

> 华为ICT认证系列丛书

华为技术认证

HCNA网络技术学习指南

华为技术有限公司 主编



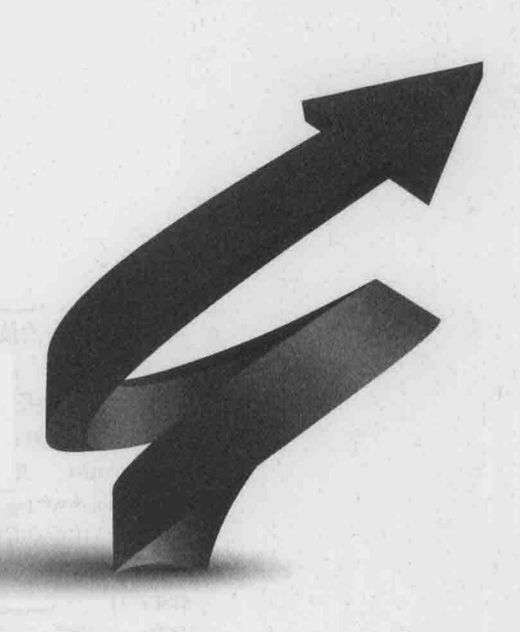




华为技术认证

HCNA网络技术学习指南

华为技术有限公司 主编



人民邮电出版社

北京

图书在版编目（CIP）数据

HCNA网络技术学习指南 / 华为技术有限公司主编
. -- 北京 : 人民邮电出版社, 2015.5
（华为ICT认证系列丛书）
ISBN 978-7-115-38634-2

Ⅰ. ①H… Ⅱ. ①华… Ⅲ. ①企业－计算机网络
Ⅳ. ①TP393.18

中国版本图书馆CIP数据核字(2015)第056754号

内 容 提 要

本书是一本配套华为HCNA认证的学习指导用书，旨在帮助读者学习并理解HCNA网络技术原理知识中的要点和难点，内容包括：网络通信基础知识、华为VRP操作系统、以太网的工作原理、STP协议、VLAN原理、IP基础知识、TCP与UDP、路由协议基础、RIP协议、OSPF协议、VLAN间的三层通信、链路聚合技术、Smart Link与Monitor Link、DHCP、地址转换技术、PPP与PPPoE、网络安全与网络管理。

2014年5月出版的《HCNA网络技术实验指南》一书是一本实验指导用书，其目的是帮助读者提升实际的动手操作能力。与之相应，本书的目的在于帮助读者对于原理性知识的理解和掌握。希望读者朋友们能够结合使用这两本书，从中获得双倍的学习效率和学习效果。

◆ 主　　编　华为技术有限公司
责任编辑　李　静
责任印制　彭志环
◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京隆昌伟业印刷有限公司印刷
◆ 开本：787×1092　1/16
印张：21　　2015年5月第1版
字数：487千字　　2015年5月北京第1次印刷

定价：59.00元

读者服务热线：(010)81055488　印装质量热线：(010)81055316
反盗版热线：(010)81055315

序

作为全球领先的信息与通信解决方案供应商，华为以“丰富人们的沟通和生活”为愿景，利用在ICT领域的专业技术和经验，帮助不同地区的人们平等、自由地接入到信息社会，确保所有人都能享受到信息和通信服务的基本权利，消除数字鸿沟。我们提倡和致力于信息和通信技术的普及，增加教育机会并培养ICT人才。

为帮助广大ICT从业人员更好地学习信息和网络技术，华为技术有限公司于2012年9月发布了业界首款免费的企业网络仿真软件平台eNSP（Enterprise Network Simulation Platform）。eNSP一经推出就受到社会的广泛关注和欢迎，下载量已超过百万，迅速成为ICT从业人员学习信息和网络技术的首选工具。2014年5月出版的、与eNSP配套使用的《HCNA网络技术实验指南》一书，更是让广大的读者朋友们体验到了利用eNSP学习信息和网络技术的高效性和趣味性。

细心的读者可能已经注意到，此次出版的《HCNA网络技术学习指南》一书，在书名上只与《HCNA网络技术实验指南》一书相差两字。正是如此，《HCNA网络技术实验指南》是一本实验指导用书，目的是提升读者朋友们对于各种网络设备和组网的实际操作能力，重在动手，重在实践；与之相应，《HCNA网络技术学习指南》旨在帮助读者朋友们对于各种网络技术原理知识的理解和掌握，重在分析，重在理论。关于理论与实践相结合之必要性，在此自不必多言。

本书主编江永红博士在华为工作已近20年，现为华为资深技术专家，且之前于国内外高校从事过多年的教学工作，对于知识的学习及传授方法有着深刻的领悟。本书是江永红博士及其写作团队的潜心专力之作，书中的各种比喻形象而生动，原理图绘制细致入微，文字描述上更是一丝不苟，几可谓字斟句酌。众所周知，对于原理性知识的阐释，除了整体上逻辑的连贯之外，最为重要的便是细节描述上的正确与严谨。本书之表现，正是对这一浅显道理的深刻体现。

总之，就HCNA网络技术知识的学习而言，相信《HCNA网络技术实验指南》和《HCNA网络技术学习指南》这两本书一定会让读者朋友们爱不释手。同时也相信，在不久的将来，ICT领域的许多拔萃之才曾经正是这两本书的忠实读者！

全球培训与认证部部长
华为企业业务集团
2015年1月

前　　言

特别声明

本书是一本配套华为HCNA认证的学习指导用书，旨在帮助读者朋友们学习并理解HCNA网络技术原理知识中的要点和难点。除了本书所涵盖的要点和难点知识外，HCNA还涉及了许多其他的知识点，如：RSTP、MSTP、DNS、FTP、VRRP、NAC、802.1x、SSH、xDSL、HDLC、FR、GRE、IPSec、WLAN、VoIP、数据中心、云计算、3G/4G、IPv6等。对于希望考取HCNA认证的读者朋友，除了应熟练地掌握本书所涵盖的要点和难点知识外，还应对其他的各个知识点有一个基本的了解。

本书内容组织

本书采用了章、节、小节三级结构，分别对应了一级、二级、三级目录。本书共分13章，其中第1和第2章为本书的铺垫性内容，第3～13章为本书的核心内容。最后为附录，附录中给出了书中所有练习题的答案。

第1章：网络通信基础

学习网络通信技术知识，必须首先了解OSI和TCP/IP这两种基础模型。本章对这两种基础模型进行了比较详细的介绍，重点对它们之间的差异性进行了描述。本章还对网络的典型拓扑形态、局域网与广域网、传输介质、通信方式等基础知识进行了简单的介绍。

第2章：VRP基础

本书的主要目的是帮助读者学习和理解网络通信技术的基本原理。为了减轻这些原理性知识的“抽象感”，增强“触摸感”，书中适量地加入了一些示意性的配置实验内容；学习这些配置实验内容之前，必须对VRP（Versatile Routing Platform）有一个基本的了解。VRP是华为公司从低端到高端的全系列路由器、交换机等数据通信产品的通用网络操作系统，本章对于VRP的基本使用方法进行了系统的介绍。学习好本章的知识内容，是掌握华为产品技术的基本前提。

第3章：以太网

在当今的网络通信技术领域中，以太网的重要性是不言而喻的，它也几乎成为了局域网的代名词。本章以计算机及交换机上的以太网卡为切入点，系统而详细地描述了关于以太网的原理性知识，这些知识的关键点包括：计算机上的以太网卡与交换机上的以太网卡的异同点，MAC地址的结构和分类，以太帧的结构和分类，交换机的转发原理及MAC地址表的生成过程和动态属性，ARP的工作原理。

第4章：STP协议

由计算机和交换机组成的以太网所面临的一个主要问题是二层环路的问题，解决这

一问题的根本方法是在交换机上运行 STP 协议或 STP 协议的改进型协议（如 RSTP、MSTP 等）。本章系统而详细地描述了 STP 协议的产生原因及工作原理和过程；对于其他改进型防环协议，本章未做描述。

第 5 章：VLAN

由计算机和交换机组成的以太网所面临的另一个主要问题是如何灵活而有效地划分二层广播域，通用的方法是采用 VLAN 技术。本章系统而详细地描述了 VLAN 的基本原理、VLAN 帧的格式、VLAN 的链路类型和端口类型、VLAN 帧的转发过程。本章还对 GVRP 协议的功能作用进行了说明。

第 6 章：IP 基础

第 3、第 4 和第 5 章都只涉及数据链路层的知识，本章开始介绍 IP 层面的基础知识，主要包括 IP 编址、IP 报文格式、IP 转发这 3 方面的内容。学习完本章内容之后，读者应该对二层通信、三层通信、internet、Internet 等诸多容易混淆的基本概念有一个清晰的认识。

第 7 章：TCP 与 UDP

本章对传输层的 TCP 和 UDP 进行了简单的描述，重点是对无连接的通信方式与面向连接的通信方式进行了比较分析，同时也着重分析了 TCP 会话的建立过程以及 TCP 的确认与重传机制。

第 8 章：路由协议基础

路由协议一向被认为是 IP 网络知识的难点内容，本章的目的是希望帮助读者在此方面打下一个坚实的基础。本章首先讲述了路由的含义、路由的来源、路由的优先级、路由的开销、静态路由、动态路由、默认路由、路由表等基本概念，然后选择了 RIP 这种最为简单的路由协议进行了全面而深入的分析。对于 OSPF 协议，本章只对其进行了概念性的描述。本章未涉及 IS-IS、BGP 等其他路由协议的内容。

第 9 章：VLAN 间的三层通信

不同的 VLAN 之间虽然无法实现二层通信，但是仍然可以实现三层通信。本章描述了实现 VLAN 之间三层通信的几种方法：多臂路由器方法、单臂路由器方法和三层交换机方法。本章的重点是关于三层交换机在数据转发原理方面与二层交换机和传统路由器的差异性。

第 10 章：链路技术

链路聚合是一种常见的链路技术，它可以灵活地增加设备之间的连接带宽，同时又可增强设备之间连接的可靠性。本章详细地描述了链路聚合的基本概念、适用场景、原理过程。本章还对 Smart Link 和 Monitor Link 这两种华为专有的、用以增强网络连接可靠性的链路技术进行了介绍和说明。

第 11 章：DHCP 及网络地址转换技术

本章描述了 DHCP 的基本概念和工作流程，同时也对 DHCP 中继进行了简单的介绍。本章还对网络地址转换技术的基本概念、基本原理、使用场景进行了适当的描述和分析。

第 12 章：PPP 与 PPPoE

本书只涉及两种典型的数据链路层技术：一种是以太网技术；另一种是 PPP 技术。本章首先对 PPP 技术的基本概念、PPP 帧的格式、PPP 的工作流程、PPP 的主要工作阶

段进行了系统的描述和分析，然后对 PPP 技术与以太网技术的结合体——PPPoE 进行了适当的描述和分析。

第 13 章：网络安全与网络管理

网络安全与网络管理是网络技术领域中的两个特别重要的分支，但本章只是简单地触碰了一下这两方面的内容。本章首先讲解了经常用于网络安全中的 ACL 技术，然后对网络管理系统中所使用的 3 个基本协议（SMI、MIB、SNMP）进行了简单的介绍。

附录：练习题答案

为了帮助读者朋友们自检学习效果，本书的许多章节都留有适量的练习题。本章给出了所有这些练习题的答案。

适用读者对象

本书的基本定位是一本配套华为 HCNA 认证的学习指导用书，特别适合于学习和备考 HCNA 的读者朋友。本书对于路由交换基础知识中的要点和难点进行了非常详细而透彻的讲解，对于希望准确而深刻地理解路由交换原理知识的高校学生、ICT 从业人员以及网络技术爱好者，阅读本书无疑是一种很好的选择。

阅读注意事项

读者在阅读本书的过程中，需要注意以下事项。

1. 对于超出本书知识范围或难度的知识点，书中会以“不做介绍”、“不做描述”、“不做分析”、“不做讨论”、“不做深究”、“不做解释”等文字进行明确的提示。是否掌握了这些知识点，几乎不会影响到对于本书知识内容的阅读理解。

2. 本书中的以太网均是指星型以太网。对于早期的总线型以太网以及与总线型以太网紧密相关的 CSMA/CD（Carrier Sense Multiple Access with Collision Detection）协议和冲突域等概念，本书未做任何描述和分析。对以太网的发展历史感兴趣的读者，可以自行查阅相关资料以做了解。需要说明的是，不了解这些“历史知识”，完全不会影响到对于本书知识内容的阅读理解。

3. 本书未涉及关于 IPv6 的任何具体描述。若无特别说明，本书中的 IP 一律是指 IPv4。

4. 关于数据链路层技术的具体描述和分析，本书只涉及了以太网技术和 PPP 技术。若无特别说明，书中的“网卡”、“网口”、“接口”、“端口”等等默认是指“以太网卡”、“以太网口”、“以太接口”、“以太端口”等；若无特别说明，书中的“帧”默认是指“以太帧”。

5. 本书习惯上把路由器或计算机上的网口称为“接口”，把交换机上的网口称为“端口”，这种差异仅仅是称谓习惯上的差异。在平时的交流中，“接口”一词与“端口”一词完全可以混用。

6. 若无特别说明，本书中的“交换机”一词默认是指不具备三层转发功能的以太网二层交换机。

7. 本书 8.1.2 小节（路由信息的来源）中描述静态路由配置时包含了这样的陈述：静态路由的 Cost 的值可以人为地设定为 0，也可以是其他我们希望的值。这种说法在理论上总是成立的，但实际上许多网络设备商在实现静态路由配置时，Cost 的值总是规定

为固定值 0，而不允许进行任意的设定或修改。

8．许多网络设备商在实现 RIP 协议时，规定最小的 RIP 跳数为 0，即路由器到达其直连网络的 RIP 跳数为 0。但是，RIP 标准协议中规定的最小 RIP 跳数却是 1，即路由器到达其直连网络的 RIP 跳数为 1。这种差异是历史的原因造成的，但这种差异本身并不会影响到 RIP 的实际部署和正常工作。本书 8.2.1 至 8.2.7 小节中，最小 RIP 跳数的定义是 1；8.2.8 小节（RIP 基本配置示例）中，最小 RIP 跳数的定义是 0。

9．对于本书的任何意见和建议，敬请发送邮件至 Learning@huawei.com。

本书常用图标



以下是参与本书编写人员的名单（排名不分先后）

主　　编：江永红

编委人员：陈　哲、吴　平、胡园园、霍迪雅、周剑毫、姚成霞、白　杰、刘怀毅、王琳琢、周　欢、戚鹏飞、宗　悦、王超伟、叶　涛、秦章伟、陈　莹、付　海、王晓露、苏　蒙、张梦实、王振科、赵芳芳、高继国、李　丽、张宜清、张　超、马　骞、屠晓峰、徐一鸣、刘　洋

华为认证简介

华为认证是华为公司凭借多年信息通信技术人才培养经验，以及对行业发展的深刻理解，基于ICT（Information Communication Technology，信息通信技术）产业链人才个人职业发展生命周期，搭载华为“云—管—端”融合技术，推出的覆盖IP、IT、CT以及ICT 融合技术领域的认证体系，是业界唯一的ICT 全技术领域认证体系。

华为技术有限公司经过20多年在ICT行业培训和认证领域的积累，已经在全球形成了完整的培训认证体系，包括自有的培训中心、授权的培训中心以及与高校合作的教育项目，累计参加华为培训的人次已超过300万，培训与考试服务覆盖160多个国家。

对行业不同领域的人才，华为均有与之匹配的知识和技能培养解决方案，对其进行准确合理的能力评估。针对个人的职业发展历程，华为提供从工程师到资深工程师、专家、架构师层级，以及从单一的技术领域到ICT融合的职业技术认证体系。

如果希望全面了解华为认证培训的相关信息，敬请访问华为培训认证主页（http://support.huawei.com/learning）；如果希望了解华为认证最新动态，敬请关注华为认证官方微博（http://e.weibo.com/hwcertification）；如果希望和广大用户一起进行技术问题的探讨，以及考试学习资料的分享，可通过华为官方论坛链接（http://support.huawei.com/ecommunity/bbs）点击进入华为认证版块。华为职业技术认证包含的内容如图1所示。

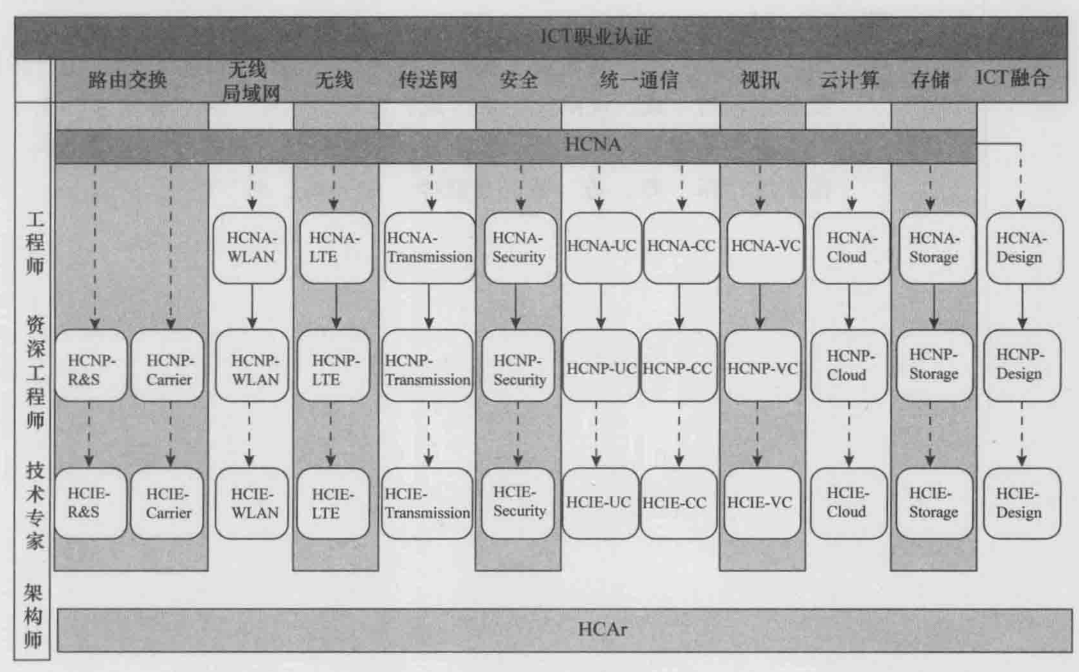
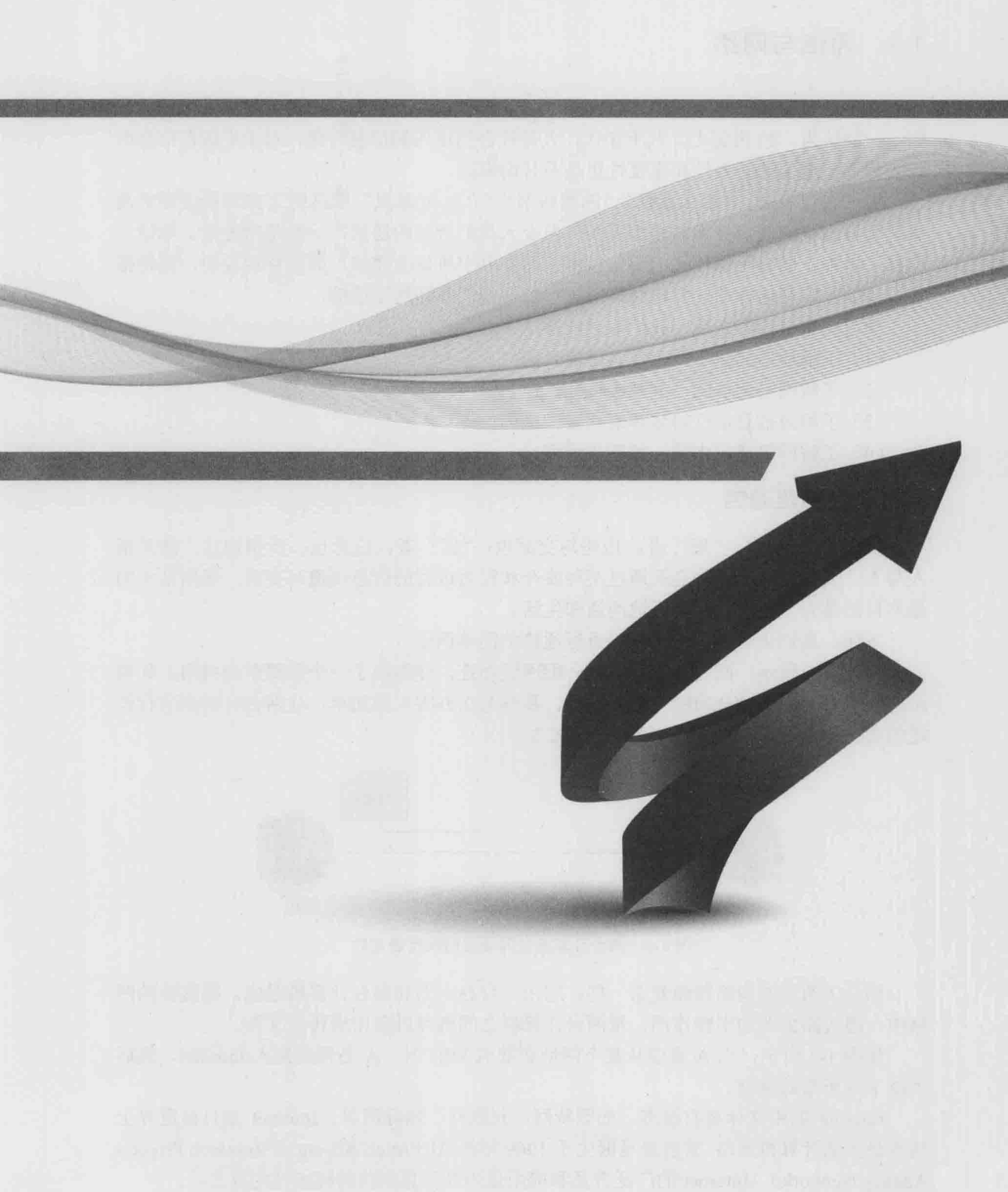


图1　华为职业技术认证的内容

第1章
网络通信基础

目　录



1.1 通信与网络

通信的概念我们并不陌生，在人类社会的起源和发展过程中，通信就一直伴随着我们。一般认为，20 世纪七、八十年代，人类社会已进入到信息时代，对于生活在信息时代的我们，通信的必要性和重要性更是不言而喻的。

通信古已有之，“烽火戏侯”“鸿雁传书”“八百里急报”等这些大家耳熟能详的故事就是与古代的通信技术紧密相关的。但今天我们所说的通信，一般是指电报、电话、广播、电视、网络等现代化的通信技术。而本书中所说的通信，如无特别说明，则是指通过诸如互联网这样的计算机网络所进行的通信，亦即网络通信。

学习完本节内容之后，我们应该能够：

（1）了解通信的基本概念及其终极目的；

（2）了解网络通信的一些基本特征；

（3）了解对信息进行封装和解封装的原因；

（4）了解网络通信中的一些常见术语。

1.1.1 什么是通信

“通信”一词中，“通”者，传递与交流也；“信”者，信息也。所谓通信，就是指人与人、人与物、物与物之间通过某种媒介和行为进行的信息传递与交流。通信技术的最终目的是为了帮助人们更好地沟通和生活。

下面，我们来看几个通过网络进行通信的简单例子。

如图 1-1 所示，两台计算机通过一根网线相连，便组成了一个最简单的网络。如果 A 想从 B 那里获得“B.MP3”这首歌曲，那该怎么办呢？很简单，让两台计算机运行合适的文件传输软件并单击几下鼠标就 OK 了。

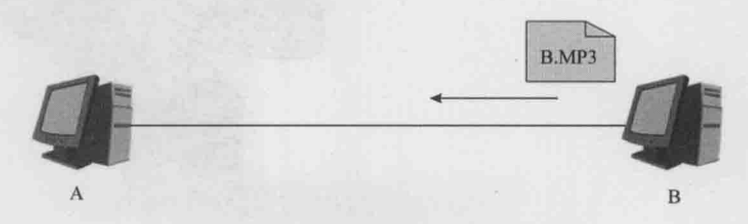


图 1-1 两台计算机之间通过网线传递文件

图 1-2 所示的网络稍微复杂一些，它由一台路由器和多台计算机组成。在这样的网络中，通过路由器的中转作用，每两台计算机之间都可以自由地传递文件。

如图 1-3 所示，当 A 希望从某个网址获取 A.MP3 时，A 必须先接入 Internet，然后才能下载所需的歌曲。

Internet 的中文译名有很多，如因特网、互联网、网际网等。Internet 是目前世界上规模最大的计算机网络，其前身是诞生于 1969 年的 ARPAnet（Advanced Research Projects Agency Network）。Internet 的广泛普及和应用是当今信息时代的标志性内容之一。

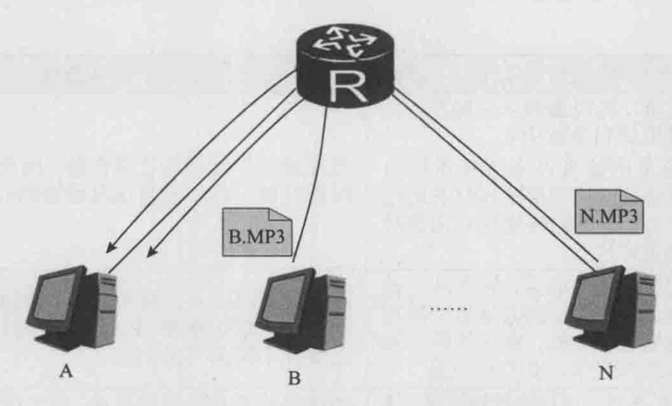


图 1-2　多台计算机通过路由器传递文件

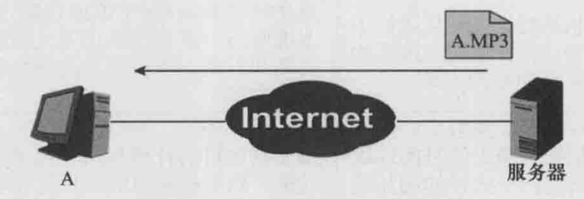


图 1-3　通过 Internet 下载文件

1.1.2　快递与网络通信

为了形象地理解信息这种虚拟存在的传递过程，我们先了解一下日常生活中经常会接触到的物品快递服务。图 1-4 示意了从某省会城市到另一省会城市的物品快递的各个主要步骤。

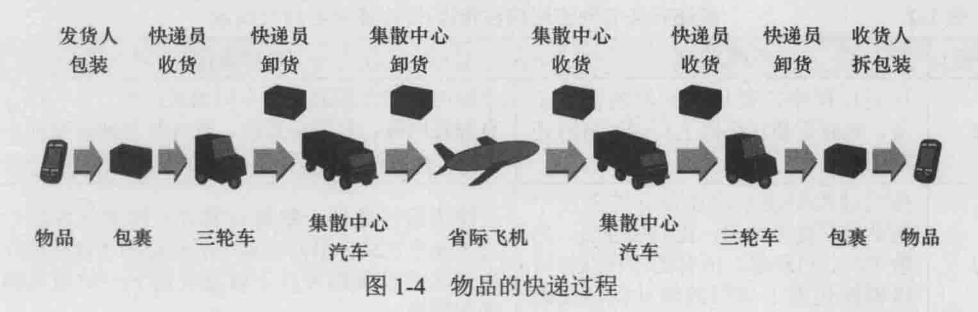


图 1-4　物品的快递过程

虚拟信息的传递过程与真实物品的传递过程有许多相似之处，表 1-1 给出了二者之间的对比。在比较的过程中，我们引入了一些网络通信的常用术语。

表 1-1　快递过程与网络通信过程的对比

序号	物品快递	网络通信
1	发货人准备好需要快递的物品	应用程序生成需要传递的信息（或称数据）
2	发货人使用纸盒将物品包装起来	应用程序将数据打包成原始的“数据载荷”
3	发货人将纸盒装入快递员提供的塑料袋，填写快递单并粘贴在袋子表面，形成包裹 快递单上最重要的内容是收货人的姓名和地址	在原始的数据载荷的前后分别加上“头部”和“尾部”，形成“报文”。报文头部中最重要的信息是接收者的地址信息，亦即“目的地址” 在一个信息单元的基础上，增加一些新的信息段，使其形成一个新的信息单元，这个过程称为“封装”

（续表）

序号	物品快递	网络通信
4	快递员收货，将包裹装入三轮车，并通过公路运送到集散中心 虽然三轮车中装有许多去往不同目的地的包裹，但是三轮车只负责运送到集散中心。剩下的传递过程由集散中心继续负责进行	报文通过网线到达计算机的“网关”。网线所起的作用跟公路一样，它是信息传输的介质
5	三轮车到达集散中心后，包裹从三轮车中取出。集散中心根据包裹上填写的目的地址进行分检，将去往同一城市的物品装入汽车，并开往机场	网关收到报文后，对其进行“解封装”，读取其目的地址，然后再重新封装，并根据目的地址的不同决定发往不同的“路由器”
6	汽车到达机场后，进行卸货和装货。飞机装载着去往同一城市的包裹飞向天空	通过网关及路由器的传递，报文最终离开本地网络，进入 Internet 的干道进行传输
7	飞机抵达目的机场后，包裹从飞机中取出并进行分检，将去往同一地区的包裹装入相应集散中心的汽车	报文经过 Internet 干道的传输，到达目的地址所在的本地网络。本地网络的网关或路由器对报文进行解封装和封装，并根据目的地址决定发往相应的下一台路由器
8	汽车到达集散中心后，集散中心先进行卸货，然后根据包裹上的目的地址进行分检，将去往同一大楼的物品装入快递员的三轮车	报文到达目的计算机所在网络的网关，被解封装和封装，然后根据目的地址发往相应的计算机
9	快递员送货上门。收货人拆开塑料袋及纸盒，确认物品完好无损后收下。整个快递过程完成	计算机接收到报文后，对报文进行校验处理。校验无误后，接收下报文，并将其中的数据载荷交由相应的应用程序进行处理。一次完整的网络通信过程便告结束

通过比较，我们可以发现快递服务与网络通信在传递/传输技术上有许多共同点，如表 1-2 所示。

表 1-2　　快递服务与网络通信在传递/传输技术上的共同点

序号	物品快递	网络通信
1	传递过程中需要用到不同的交通工具，有时需要在公路上传递，有时还需要在空中传递	不同的传输介质适合于不同的通信场合，传输介质有时是网线，有时是光纤，有时甚至就是空间本身 但是，传输介质的变化不会改变需要传递的信息本身
2	传递过程中通常需要设立若干个中转站进行接力传递，比如快递员、集散中心、机场等。所有的中转站都可以根据包裹上填写的地址找到正确的传递方向	远距离网络通信一般都需要多个网络设备进行接力传输来完成。根据信息中所标记的“目的地址”，所有网络设备都可以正确地决定下一步该向哪里进行传输
3	在整个传递的过程中，物品需要包装和拆包装，包裹需要多次装载和卸货。这些操作类似于对信息的封装和解封装	对信息进行多次封装和解封装，主要有两个目的： • 给信息补充一些标记，帮助网络设备正确判断该如何对信息进行处理和传输（就像粘贴快递单一样） • 为了适应不同的传输介质和传输协议（就像要先将包裹从三轮车中卸下来，再装入汽车一样，而不能直接将三轮车全部塞入汽车）
4	每一个中转站都只需要关注自己所负责的那一段路程，而无需关注包裹的整个传递过程 比如，快递员只需要负责将包裹正确传递到集散中心即可，无需关心后面如何处理包裹	各个网络设备都只需要完成信息在某一段距离范围内的正确传输。上一站只需要负责将信息传递给正确的下一站，下一站只需要将信息传递给正确的下下一站，如此等等

至此，我们通过快递服务这个比喻，粗略地认识了一下网络通信的一些基本特征。采用快递服务这样的比喻，仅仅是便于大家对网络通信有一个直观而形象的认识。读者不必纠结于二者在对比过程中的一些细节问题，因为比喻毕竟是比喻，网络通信与快递服务之间的许多细节特征无法满足精确的一一对应关系。

1.1.3 一些常见的术语

我们在 1.1.2 小节中引出了一些常见的网络通信术语，表 1-3 给出了对这些术语的进一步解释和说明。

表 1-3 常见术语说明

术语	解释和说明
数据载荷	根据快递服务的比喻，我们将数据载荷理解为最终想要传递的信息。而实际上，在具有层次化结构的网络通信过程中，上一层协议传递给下一层协议的数据单元（报文）都可以称之为下一层协议的数据载荷
报文	报文是网络中交换与传输的数据单元，它具有一定的内在格式，并通常都具有头部+数据载荷+尾部的基本结构。在传输过程中，报文的格式和内容可能会发生改变
头部	为了更好地传递信息，在组装报文时，在数据载荷的前面添加的信息段统称为报文的头部
尾部	为了更好地传递信息，在组装报文时，在数据载荷的后面添加的信息段统称为报文的尾部。注意，很多报文是没有尾部的
封装	对数据载荷添加头部和尾部，从而形成新的报文的过程
解封装	解封装是封装的逆过程，也就是去掉报文的头部和尾部，获取数据载荷的过程
网关	网关是在采用不同体系结构或协议的网络之间进行互通时，用于提供协议转换、路由选择、数据交换等功能的网络设备。网关是一种根据其部署位置和功能而命名的术语，而不是一种特定的设备类型
路由器	简单地讲，路由器就是为报文选择传递路径的网络设备。后续的学习会使读者对路由器的理解更加深入而全面

1.1.4 练习题

1. （多选）以下哪些是网络通信的例子？（ ）
   A．使用即时通信软件（如 QQ）与好友聊天
   B．使用计算机在线观看视频
   C．从公司的邮箱下载邮件到自己的电脑中
   D．使用传统座机进行通话
2. （多选）以下哪些与封装和解封装的目的有关？（ ）
   A．加快通信的速度　　B．不同网络之间的互通
   C．通信协议的分层　　D．缩短报文的长度

1.2 OSI 模型和 TCP/IP 模型

OSI 模型和 TCP/IP 模型几乎应该算是网络通信领域中使用频率最高的两个术语了。

特别是 TCP/IP，它几乎成了网络通信的代名词。从整体上先大致了解一下 OSI 模型和 TCP/IP 模型，对于我们接下来具体而深入地学习各种网络通信知识，具有非常重要的指导性作用。

在学习本节内容的过程中，请特别注意以下几点。

（1）网络协议的定义和作用。

（2）网络分层（功能分层，协议分层）的理念。

（3）OSI 模型与 TCP/IP 模型的异同点。

学习完本节内容之后，我们应该能够：

（1）说出几个网络协议的名称以及几个知名的标准机构；

（2）大致描述 OSI 模型和 TCP/IP 模型的基本内容。

1.2.1 网络协议和标准机构

我们通常使用汉语、英语、法语等这样的自然语言进行思想的沟通和交流。其实，从网络通信的角度来看，这些各种各样的自然语言就相当于是网络通信中所使用的各种各样的通信协议。譬如，某人说："I 服了 You！你也太 out 了吧，连屌丝和沙发这些词是什么意思都不懂！"我们便可以认为这句话中涉及了汉语这个"协议"和英语这个"协议"。更进一步讲，屌丝、沙发这些词是属于汉语中新兴出现的"网络语言"，这些"网络语言"可以被认为是汉语这个"协议"中的"子协议"。听这句话的人需要懂得汉语、英语，以及网络语言，才能真正明白对方的意思。

在网络通信中，所谓协议，就是指诸如计算机、交换机、路由器等网络设备为了实现通信而必须遵从的、事先定义好的一系列规则和约定。我们经常提到的 HTTP（Hypertext Transfer Protocol）、FTP（File Transfer Protocol）、TCP（Transmission Control Protocol）、IPv4、IEEE 802.3（以太网协议）等，都是网络通信协议的例子。譬如，当我们通过浏览器访问网站时，网址中的"http://"就说明这次访问会使用 HTTP。我们通过 FTP 工具下载文件时，文件地址中的"ftp://"就说明这次下载会使用 FTP。

需要特别说明的是，在网络通信领域中，"协议"、"标准"、"规范"、"技术"等这些词汇是经常混用的。譬如，IEEE 802.3 协议、IEEE 802.3 标准、IEEE 802.3 协议规范、IEEE 802.3 协议标准、IEEE 802.3 标准协议、IEEE 802.3 标准规范、IEEE 802.3 技术规范等，说的都是一回事。

协议可分为两类，一类是各网络设备厂商自己定义的私有协议，另一类是专门的标准机构定义的开放式协议（或称开放性协议，开放协议），二者的关系有点像方言与普通话的关系。显然，为了促进网络的普遍性互联，各厂商应尽量遵从开放式协议，减少私有协议的使用。

专门整理、研究、制定和发布开放性标准协议的组织称为标准机构。表 1-4 列出了几个在网络通信领域非常知名的标准机构。

表 1-4 知名网络标准机构

标准机构	说明
国际标准化组织（International Organization for Standardization，ISO）	ISO 是世界上最大的非政府性标准化专门机构，是国际标准化领域中一个十分重要的组织。ISO 的任务是促进全球范围内的标准化及其有关活动，以利于国际间产品与服务的交流，以及在知识、科学、技术和经济活动中发展国际间的相互合作
互联网工程任务组（Internet Engineering Task Force，IETF）	IETF 是全球互联网最具权威的技术标准化组织，其主要任务是负责互联网相关技术规范的研发和制定。目前，绝大多数的互联网技术标准都是出自 IETF。著名的 RFC（Request For Comments）标准系列就是由 IETF 制定和发布的
电气电子工程师学会（Institute of Electrical and Electronics Engineers，IEEE）	IEEE 是世界上最大的专业技术组织之一。IEEE 成立的目的在于为电气电子方面的科学家、工程师、制造商提供国际联络交流的场合，并为他们提供专业教育、提高专业能力服务。著名的以太网标准规范就是 IEEE 的杰作之一
国际电信联盟（International Telecommunications Union，ITU）	ITU 是主管信息通信技术事务的联合国机构，也简称为“国际电联”或“电联”
电子工业联盟（Electronic Industries Alliance，EIA）	EIA 是美国电子行业标准制定者之一，常见的 RS-232 串口标准便是由 EIA 制定的
国际电工技术委员会（International Electrotechnical Commission，IEC）	IEC 主要负责有关电气工程和电子工程领域中的国际标准化工作。该组织与 ISO、ITU、IEEE 等有着非常紧密的合作关系

1.2.2 OSI 参考模型

为了实现网络的互通以及各种各样的网络应用，网络设备需要运行各种各样的协议以实现各种各样的具体的功能。面对各种各样且数量繁多的功能，我们可以从网络架构的角度引入功能分层的模型：对各种各样的具体的功能进行分门别类，将具有相似或相近目的和作用的一些功能划归到同一层面；属于同一层面的不同功能，其目的和作用是相似或相近的；属于不同层面的功能，其目的和作用是具有明显的差异的。

我们知道，网络设备的各种功能是通过运行各种相应的协议而实现的。因此，与功能分层模型相对应的便是协议分层模型：属于同一层面的不同协议，其功能作用是相似或相近的；属于不同层面的协议，其功能作用具有明显的差异。

建立层次化的协议（或功能）模型带来的主要好处有以下几点。

（1）更易于标准化：每一层都聚焦于自己所在层面的主要功能，不至于使问题发散，这样就更容易制定出相应的协议或标准。

（2）降低关联性：某一层协议的增加、减少、更新或变化，不至于影响到其他层面协议的工作；各层协议可以相对独立地自由发展。

（3）更易于学习和理解：对于学习和研究网络的人员来说，分层模型可以使得整个网络的工作机制以及众多网络协议之间的关系更加清楚明晰，易于学习和理解。

20 世纪 80 年代，ISO 提出了著名的开放系统互连参考模型（Open Systems Interconnection Reference Model，OSI-RM）。OSI 参考模型的出现，极大地推动了网络技术的发展。

OSI 是一个 7 层功能/协议模型，表 1-5 给出了对它的简单描述。

表 1-5　**OSI 参考模型各层功能**

层编号	层名	主要功能
1	物理层	完成逻辑上的“0”和“1”向适合于传输介质承载的物理信号（光/电信号）的转换；实现物理信号的发送、接收，以及在介质上的传输过程
2	数据链路层	在通过物理链路相连接的相邻节点之间，建立逻辑意义上的数据链路，在数据链路上实现数据的点到点或点到多点方式的直接通信
3	网络层	根据数据中包含的网络层地址信息，实现数据从任何一个节点到任何另外一个节点的整个传输过程
4	传输层	建立、维护和取消一次端到端的数据传输过程，控制传输节奏的快慢，调整数据的排序等
5	会话层	在通信双方之间建立、管理和终止会话，确定双方是否应该开始进行某一方发起的通信等
6	表示层	进行数据格式的转换，以确保一个系统生成的应用层数据能够被另外一个系统的应用层所识别和理解
7	应用层	向用户应用软件提供丰富的系统应用接口

对 OSI 参考模型中各层功能的进一步补充解释如下。

（1）物理层实现了逻辑上的数据与可以感知和测量的光/电信号之间的转换。物理层功能是通信过程的基础。物理层关注的是单个“0”和“1”的发送、传输和接收。

（2）数据链路层实现了有内在结构和意义的一连串的“0”和“1”的发送和接收。如果没有数据链路层，则通信的双方只能看到不断变化的光/电信号，并从中识别出一连串的“0”和“1”，但却不能将这些“0”和“1”组织起来，形成有意义、可理解的数据。

（3）数据链路层实现的是数据在相邻节点之间的（这里的“相邻节点”是指其间不跨越任何路由节点）、局部性的直接传递，局域网技术便是聚焦在数据链路层及其下面的物理层。而网络层需要实现的则是任意两个节点之间的、全局性的数据传递。

（4）两个人在谈话交流时，如果一个人说得太快，另一个人通常会说：“你说慢点。”“你说慢点”这句话的作用其实是在控制谈话交流的速度。如果一个人在听对方说话时，有的话没有听清楚，通常就会说：“对不起，刚才没听清楚，你再说一遍。”“……你再说一遍”这句话其实是在提高谈话交流的可靠性。传输层的某些功能非常类似于“你说慢点”“你说快点”“请再说一遍”等起的作用。

（5）我们上网请求某种网络服务时，由于输错了账号/密码，结果服务请求被拒绝。服务提供方对我们输入的账号/密码进行了验证，发现有问题，于是立即终止了接下来的通信过程。服务提供方进行的账号/密码验证并关闭通信过程的操作，便是会话层的功能之一。

（6）我们平时常用的 rar 压缩解压工具所起的作用，就是表示层的典型功能之一。文件发送方为了减少对网络带宽资源的使用，将原始文件进行了压缩后再进行发送。如果接收方不对收到的压缩文件进行解压，就无法识别和理解所发送的原始文件的真正内容。总之，表示层的作用就是使得通信双方的应用层能够识别和理解对方应用层发送过来的数据。

（7）OSI 模型中的应用层（第七层），其实是指“系统应用层”。在“系统应用层”之上，其实还有一层（第八层），称为“用户应用层（User-defined Application Layer）”，但是

“用户应用层”已经不属于 OSI 模型的范畴。HTTP、SMTP（Simple Mail Transfer Protocol）、FTP、SNMP（Simple Network Management Protocol）等协议模块本是属于 TCP/IP 协议簇的（下面马上会讲到 TCP/IP），如果我们把这些协议模块看成是属于 OSI 模型的协议模块的话，那么这些协议模块就位于 OSI 的“系统应用层”。而像 Netscape、IE（Internet Explorer）等这些不同的网络浏览器软件就位于 OSI 的“用户应用层”，但它们都会调用“系统应用层”中的 HTTP 模块；像 Foxmail、Outlook 等这些不同的 E-mail 收发软件也位于 OSI 的“用户应用层”，但它们都会调用“系统应用层”中的 SMTP 模块。

从 OSI 模型的观点来看，计算机发送数据时，数据会从高层向底层逐层传递，在传递过程中进行相应的封装，并最终通过物理层转换为光/电信号发送出去。计算机接收数据时，数据会从底层向高层逐层传递，在传递过程中进行相应的解封装。图 1-5 示意了两台计算机和一根网线组成的简单网络中，计算机 A 向计算机 B 传递数据时的层次化处理过程。

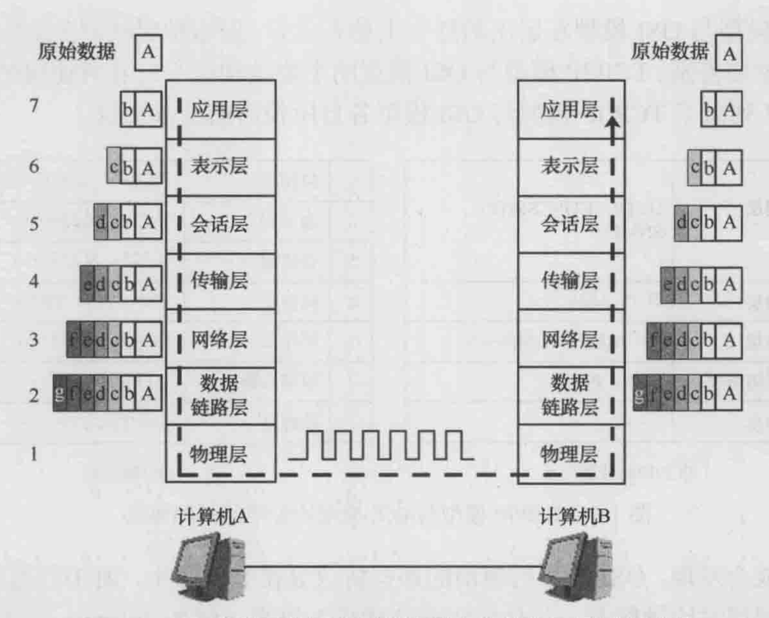


图 1-5　OSI 模型下数据在通信终端中的封装和解封装过程

1.2.3　TCP/IP 协议簇

TCP/IP 模型发端于 ARPAnet 的设计和实现，其后被 IETF 不断地充实和完善。TCP/IP 模型、TCP/IP 功能模型、TCP/IP 协议模型、TCP/IP 协议簇、TCP/IP 协议栈等说法在现实中是经常被混用的。TCP/IP 这个名字来自于这个协议簇中两个非常重要的协议，一个是 IP（Internet Protocol），另一个是 TCP（Transmission Control Protocol）。

图 1-6 给出了 TCP/IP 模型的两个不同版本，以及它们与 OSI 模型的比较。TCP/IP 标准模型共有 4 层，其“网络接入层”对应了 OSI 模型的第一层和第二层（或 TCP/IP 对等模型的第一层和第二层）。OSI 模型中的第五、六、七层的功能全部影射到了 TCP/IP 标准模型或对等模型中的应用层。现实中，五层的 TCP/IP 对等模型使用最为广泛。本书中，如无特别说明，我们所说的 TCP/IP 模型均是指 TCP/IP 对等模型。

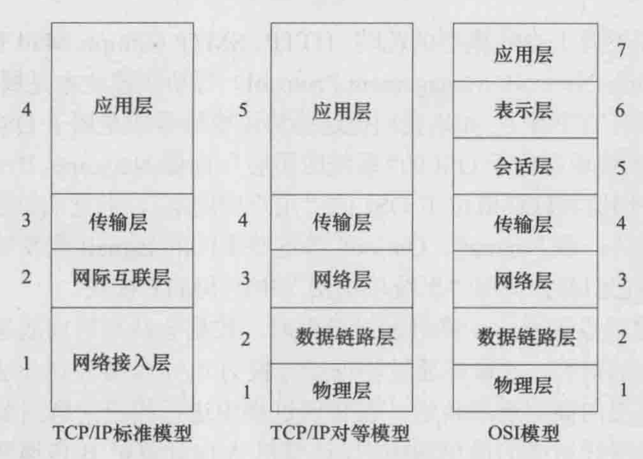


图 1-6 TCP/IP 模型的分层结构

TCP/IP 模型与 OSI 模型在层次的划分上稍有差异，但这种层次划分上的差异并不是二者之间的主要差别。TCP/IP 模型与 OSI 模型的主要差别在于二者所使用的具体协议的不同。图 1-7 列出了 TCP/IP 模型与 OSI 模型各自所使用的部分协议。

5	应用层	HTTP、FTP、SMTP、SNMP……
4	传输层	TCP、UDP……
3	网络层	IP、ICMP、IGMP……
2	数据链路层	SLIP、PPP……
1	物理层	……

TCP/IP协议簇

7	应用层	FTAM、X.400、CMIS……
6	表示层	X.226、X.236……
5	会话层	X.225、X.235……
4	传输层	TP0、TP1、TP2……
3	网络层	CLNP、X.233……
2	数据链路层	ISO/IEC 766……
1	物理层	EIA/TIA-232……

OSI协议簇

图 1-7 TCP/IP 模型与 OSI 模型所使用的不同协议

读者可能会发现，OSI 模型所使用的那些协议显得非常陌生，而 TCP/IP 模型所使用的那些协议则相对比较熟悉。为什么会是这样呢？原来，诸如 Internet 等现实中的网络的设计与实现，使用的几乎全都是 TCP/IP 协议簇，而不是 OSI 协议簇。

在 OSI 模型中，我们习惯把每一层的数据单元都称为“协议数据单元（Protocol Data Unit，PDU）”。例如，第六层的数据单元称为 L6 PDU，第三层的数据单元称为 L3 PDU，其中的 L 代表“层（Layer）”。

在 TCP/IP 模型中，我们习惯把物理层的数据单元称为“比特（Bit）”；把数据链路层的数据单元称为“帧（Frame）”；把网络层的数据单元称为“分组或包（Packet）”。对于传输层，我们习惯把通过 TCP 封装而得到的数据单元称为“段（Segment）”，即“TCP 段（TCP Segment）”；把通过 UDP 封装而得到的数据单元称为“报文（Datagram）”，即“UDP 报文（UDP Datagram）”。对于应用层，我们习惯把通过 HTTP 封装而得到的数据单元称为“HTTP 报文（HTTP Datagram）”，把通过 FTP 封装而得到的数据单元称为“FTP 报文（FTP Datagram）”，如此等等。

现在，假设我们在 Internet 上通过某网站找到了一首歌曲，并向相应的 Web 服务器请求下载这首 2 000 字节大小的歌曲，那么，这首歌曲在被发送之前将在 Web 服务器中被逐层进行封装。如图 1-8 所示，应用层会对原始歌曲数据（Data）添加 HTTP 头部，形成一个 HTTP 报文；因为该 HTTP 报文太长，所以传输层会将该 HTTP 报文分解成两部分，并在每部分前添加 TCP 头部，从而形成两个 TCP 段；网络层会对每个 TCP 段添加 IP 头部，形成 IP 包；数据链路层（假定数据链路层使用的是以太网技术）会在 IP 包的前面和后面分别添加以太网帧头和帧尾，形成以太网帧（简称以太帧）；最后，物理层会将这些以太帧转化为比特流。

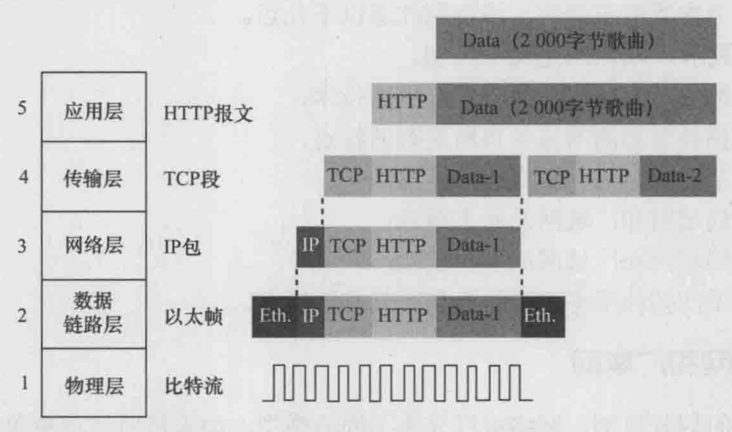


图 1-8　TCP/IP 模型中数据的封装过程

1.2.4　练习题

1. （多选）以下哪些是具体的网络协议？（　）

   A．HTTP　　B．FTP　　C．OSI　　D．ISO
   E．TCP　　F．IP　　G．TCP/IP

2. （单选）OSI 参考模型中，从下往上的层次依次是？（　）

   A．物理层→传输层→数据链路层→网络层→会话层→表示层→应用层
   B．物理层→传输层→数据链路层→网络层→会话层→应用层→表示层
   C．物理层→数据链路层→传输层→网络层→会话层→应用层→表示层
   D．物理层→数据链路层→网络层→传输层→会话层→表示层→应用层
   E．物理层→数据链路层→传输层→网络层→会话层→表示层→应用层

3. （单选）在 TCP/IP 模型中，“帧”是第几层的数据单元？（　）

   A．第一层　　B．第二层　　C．第三层
   D．第四层　　E．第五层

4. （多选）对网络协议进行分层有哪些好处？（　）

   A．有利于协议设计　　B．有利于协议管理
   C．有利于学习和理解协议　　D．有利于提高通信效率
   E．有利于修改协议

1.3 网络类型

我们常常听说局域网、广域网、私网、公网、内网、外网、电路交换网络、包交换网络、环型网、星型网、光网络等数不胜数的网络术语，它们都与网络类型有关。之所以会有这么多的网络类型，是因为在划分网络类型时可以依据各种各样不同的划分原则。

本节中，我们将依据网络的地理覆盖范围，以及网络的拓扑形态对网络进行分类，并简单了解在这两种划分原则下，各种类型的网络所具有的基本特点。

在学习本节内容的过程中，请特别注意以下几点。

（1）局域网和广域网的定义与区别。

（2）常见的局域网技术和常见的广域网技术。

（3）不同拓扑形态的网络各自所具有的特点。

学习完本节内容之后，我们应该能够：

（1）熟悉局域网和广域网的基本概念；

（2）了解局域网和广域网的基本现状；

（3）了解不同拓扑形态的网络各自所具有的特点。

1.3.1 局域网和广域网

根据不同的划分原则，网络可以分为不同的类型。如果按照地理覆盖范围来划分，则网络可以分为局域网（Local Area Network，LAN）和广域网（Wide Area Network，WAN）。表 1-6 给出了二者的比较。

表 1-6 局域网与广域网的比较

网络类型	基本特点	使用的技术
局域网	1. 覆盖范围一般在几公里之内 2. 主要作用是把分布距离较近（如：一个家庭内、一座或几座大楼内、一个校园或厂区内，等等）的若干终端电脑连接起来 3. 不会用到电信运营商的通信线路	令牌总线（Token Bus） 令牌环（Token Ring） 光纤分布式数据接口（Fiber-Distributed Data Interface，FDDI） 以太网（Ethernet） 无线局域网（Wireless LAN，WLAN） ……
广域网	1. 覆盖范围一般在几公里以上，可大至几十、几百或几千公里 2. 主要作用是把分布距离较远（如：跨越城市、跨越国家，等等）的若干局域网连接起来 3. 会用到电信运营商的通信线路	T1/E1、T3/E3 X.25 HDLC（High-level Data Link Control） PPP（Point-to-Point Protocol） ISDN（Integrated Services Digital Network） FR（Frame Relay） ATM（Asynchronous Transfer Mode） SDH（Synchronous Digital Hierarchy） ……

需要说明的是，局域网和广域网的地理覆盖范围并没有一个严格的界限。我们平时在谈论局域网或广域网时，更多的是指局域网所使用的技术或广域网所使用的技术。

我们现在经常所说的以太网和 WLAN，便是两种应用非常广泛的局域网技术。事实上，局域网技术的种类非常多，如：令牌总线（IEEE 802.4）、令牌环（IEEE 802.5）、FDDI、DQDB（Distributed Queue Dual Bus，IEEE 802.6）、isoEthernet（IEEE 802.9a）、100VG-AnyLAN（IEEE 802.12）等。然而，随着时间的推移及市场的选择，除了以太网技术和 WLAN 技术得到了积极的发展和广泛的应用外，其他所有曾经出现的各种各样的局域网技术，几乎都已经销声匿迹，或正处于被淘汰的过程中。对于这些过往的技术，我们不必再去探究其具体的原理细节（除非对网络技术发展的历史特别感兴趣）。图 1-9 简单展示了令牌总线网络和 FDDI 网络的拓扑形态，姑且算是一种怀旧吧。

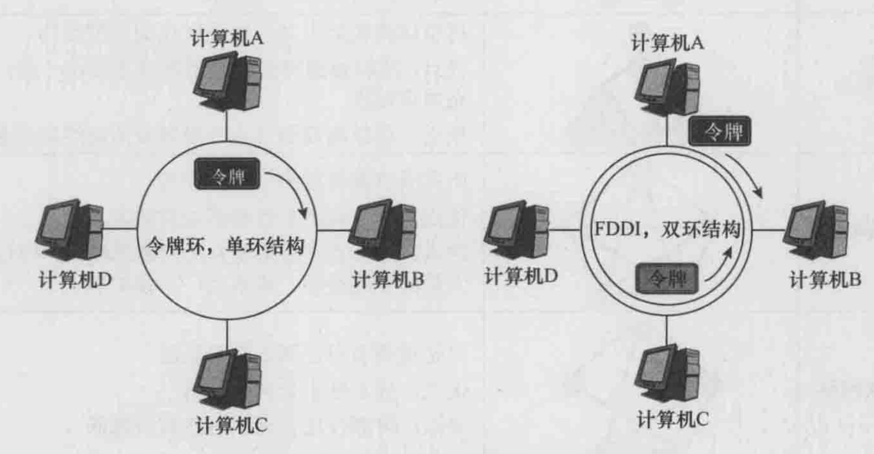


图 1-9　令牌总线网络和 FDDI 网络的拓扑形态

相对于局域网而言，广域网的部署环境和条件要复杂得多，广域网的改造和升级所涉及的因素也非常复杂，这也是为什么各种新老的广域网技术至今仍然并存的主要原因。但总的趋势是，诸如 T1/E1、T3/E3、ISDN、FR 等这些较为过时的广域网技术正在逐步地被淘汰，而像 SDH、OTN（Optical Transport Network）等这样的光网络技术已经或正在被日益广泛地应用于广域网通信的领域。

1.3.2　网络拓扑形态

除了可以依据地理覆盖范围来划分网络类型之外，我们还可以根据网络的拓扑形态来划分网络类型。网络拓扑是网络结构的一种图形化展现方式，表 1-7 给出了各种拓扑形态的网络类型。

表 1-7 中所展示的都是一些理想化的典型的网络拓扑形态。在实际组网中，通常都会根据成本、通信效率、可靠性等具体需求而采用多种拓扑形态相结合的方法。例如，图 1-10 就是环型、星型和树型的一种组合使用。

表 1-7　　各种拓扑形态的网络类型

网络类型	拓扑图	基本特点
星型网络		所有节点通过一个中心节点连接在一起 优点：很容易在网络中增加新的节点。通信数据必须经过中心节点中转，易于实现网络监控 缺点：中心节点的故障会影响到整个网络的通信
总线型网络		所有节点通过一条总线（如同轴电缆）连接在一起 优点：安装简便，节省线缆。某一节点的故障一般不会影响到整个网络的通信 缺点：总线故障会影响到整个网络的通信。某一节点发出的信息可以被所有其他节点收到，安全性低
环型网络		所有节点连成一个封闭的环形 优点：节省线缆 缺点：增加新的节点比较麻烦，必须先中断原来的环，才能插入新节点以形成新环
树型网络		树型结构实际上是一种层次化的星型结构 优点：能够快速将多个星型网络连接在一起，易于扩充网络规模 缺点：层级越高的节点故障导致的网络问题越严重
全网状网络		所有节点都通过线缆两两互连 优点：具有高可靠性和高通信效率 缺点：每个节点都需要大量的物理端口，同时还需要大量的互连线缆。成本高，不易扩展
部分网状网络		只是重要节点之间才两两互连 优点：成本低于全网状网络 缺点：可靠性比全网状网络有所降低

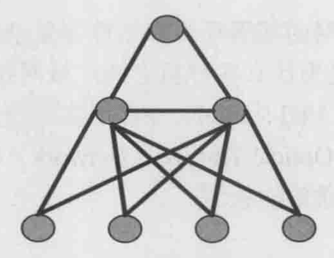


图 1-10　组合型的网络拓扑

1.3.3　练习题

1．（多选）按地理覆盖范围划分，可以将网络分为哪几种类型？（　）

A．局域网　　B．以太网　　C．互联网

D．广域网　　E．Internet

2．（单选）局域网一般限于多少范围内？（　）

A．0.1km　　B．1km　　C．10km
D．100km　　E．1000km

3．（单选）以下哪种类型的网络具有最高的可靠性？（　）
A．星型网络　　B．总线型网络　　C．环型网络
D．树型网络　　E．全网状网络

4．（单选）以下哪个特点不属于树型网络的特点？（　）
A．容易出现单点故障
B．易于扩展
C．越上层的节点可靠性要求越低

1.4 传输介质及通信方式

通信过程中所使用的物理信号必须通过某种 Medium 才能传递。Medium 一词通常翻译为“介质”“媒体”“媒介”“媒质”等。多媒体通信中也会使用到“媒体（Medium）”一词，但其中的“媒体”指的是信息的表现形式（如：声音、图像、视频、文本等）。为避免产生歧义，本书中将只使用“介质”或“传输介质”来指代光/电信号在其上进行传输的物理介质。

另外，如果有人问你：“通信方式是指什么？”你一定会觉得茫然而不知所问。“通信方式”一词的外延太广，光通信与电通信，无线通信与有线通信，单播通信与广播通信、同步通信与异步通信等，说的都是不同的通信方式。本节所说的通信方式，指的是串行通信与并行通信方式，单工、半双工以及全双工通信方式。

在学习本节内容的过程中，请特别注意以下几点。

（1）信号的物理传播速度与信息传输速率的概念性区别是什么？

（2）光信号在光纤中的物理传播速度与电信号在铜线上的物理传播速度相比较，谁更快？

（3）与铜线相比较，光纤有哪些主要优点？

（4）单模光纤与多模光纤的主要差异（结构方面、价格方面、性能方面）是什么？

（5）在以太网环境中，三类、五类、超五类 UTP 可以支持的最大信息传输率各是多少？

（6）并行通信不适于远距离通信的主要原因是什么？

学习完本节内容后，我们应该能够：

（1）了解传输介质的基本分类情况；

（2）了解网络通信中常用的传输介质的基本特性；

（3）了解并行通信与串行通信的主要区别；

（4）熟悉单工、半双工、全双工通信方式的概念和实例。

1.4.1 传输介质

现代通信技术所使用的物理信号主要是光、电信号，所使用的传输介质主要有空间、

金属导线和玻璃纤维三大类。

空间这类传输介质主要用来传递电磁波。从通信的角度来看，空间类传输介质又可分为真空和空气两种介质。电磁波在真空中的传播速度为 299 792 458m/s（也即“光速”）；电磁波在空气中的传播速度非常接近光速，大约为 299 705 000m/s。

金属导线主要用来传递电流、电压信号。在金属导线这类传输介质中，主要使用的是铜线。电流、电压信号在铜线上的传播速度也非常接近光速。网络通信中经常使用到两种结构不同的铜线，一种是同轴电缆，另一种是双绞线。

我们通常所说的“光纤”，其实就是一种玻璃纤维，它是用来传递光信号的（从本质上讲，光就是一种波长在特定范围内的电磁波）。光在光纤中的传播速度大约只有光速的 2/3，约为 200 000 000 m/s。

接下来，我们将分别简单介绍一下同轴电缆、双绞线和光纤这 3 种传输介质。

1. 同轴电缆

图 1-11 是同轴电缆（Coaxial Cable）的结构及实物外观，其中的铜导线才是用来传输电流、电压信号的，铜网屏蔽层的作用是抵御环境中的电磁辐射对所传输的电流、电压信号的干扰。有线电视网络系统广泛地使用了同轴电缆作为传输介质。早期的以太网是总线型网络，所使用的总线便是同轴电缆。目前，以太网已经演化为一种星型网络，不再使用同轴电缆，而是使用双绞线或光纤。

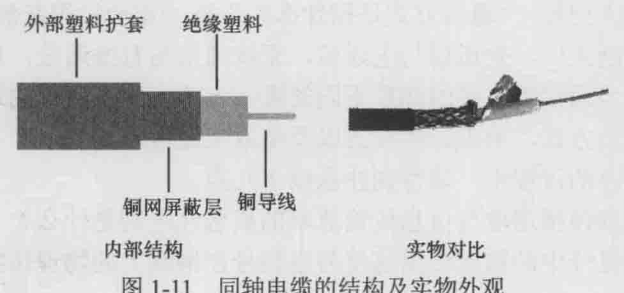


图 1-11 同轴电缆的结构及实物外观

2. 双绞线

双绞线（Twisted Pair）的名称源自于通信中所使用的铜导线通常是缠绕捆绑在一起的双绞方式。根据电磁学原理，双绞方式的导线可以较好地抵御环境电磁辐射对导线中传递的电流、电压的干扰。

依据是否包含了屏蔽层，双绞线可分为屏蔽双绞线（Shielded Twisted Pair，STP）和无屏蔽双绞线（Unshielded Twisted Pair，UTP）两种，这两种双绞线的结构和实物外观分别如图 1-12 和图 1-13 所示。从图 1-12 和图 1-13 可以看到，双绞线内的 8 根铜线两两相互缠绕，形成 4 组线对，或称 4 个绕组。显然，由于省去了屏蔽层，UTP 比 STP 要便宜一些，但是抗干扰能力也会弱一些。不过，除了某些特殊场合（如电磁辐射比较严重，或对信号传输质量要求较高等）需要使用 STP 外，一般情况下都可以使用 UTP。

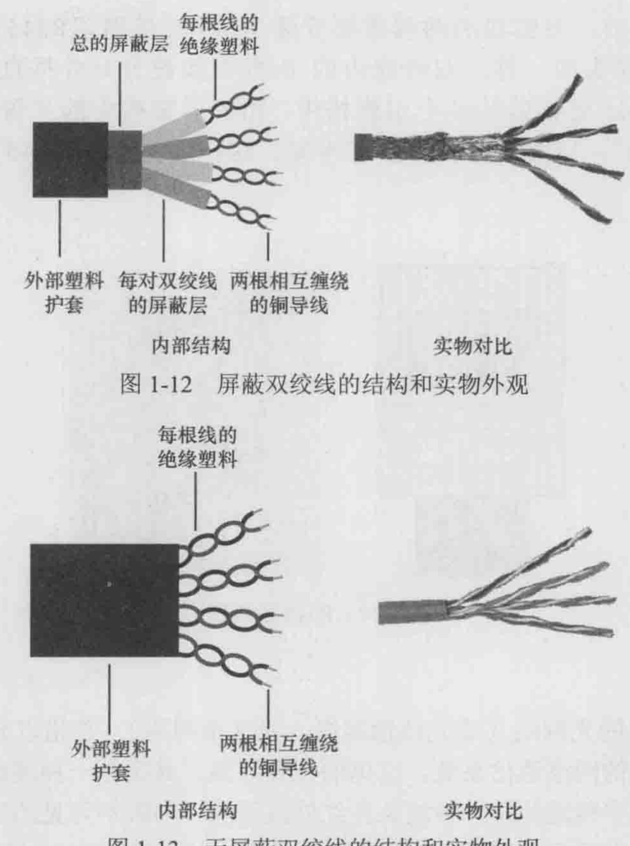


图 1-12　屏蔽双绞线的结构和实物外观

图 1-13　无屏蔽双绞线的结构和实物外观

根据材料及制作规格的不同，双绞线可以分为不同的类别，如三类双绞线、五类双绞线等，表 1-8 给出了部分 UTP 的分类情况。需要说明的是，三类双绞线、五类双绞线、超五类双绞线在应用于以太网环境时，为了保证信号在传输过程中的衰减不至于太大，其最大允许的传输距离均规定为 100m。

表 1-8　　UTP 分类

UTP	用途	说明
一类	电话系统	美国 Anixter International 公司定义，未应用在网络通信中
二类	曾用于令牌环网	美国 Anixter International 公司定义，最大信息传输速率为 4Mbit/s，现已基本不用
三类	以太网及电话系统	TIA/EIA-568 定义，最大信息传输速率为 16Mbit/s，第一个 IEEE 标准化的星型以太网标准 10Base-T 就是使用的三类 UTP
四类	曾用于令牌环网及以太网	TIA/EIA-568 定义，最大信息传输速率为 16Mbit/s，现已基本不用
五类	广泛应用于以太网及电话系统	TIA/EIA-568 定义，最大信息传输速率为 100Mbit/s，支持 10Base-T 和 100Base-TX，目前正广泛使用
超五类	广泛应用于以太网	TIA/EIA-568 定义，在五类 UTP 上进行了改进，最大信息传输速率为 1 000Mbit/s，支持 10Base-T、100Base-TX、1 000Base-T，目前正广泛使用

如图 1-14 所示，双绞线的两端需要安装 RJ45 连接器，RJ45 连接器也就是我们通常所说的水晶头的一种。双绞线内的 8 根铜线被分开并捋直后，按照一定的排序规则插入 RJ45 连接器的 8 个引脚槽中，相应引脚槽中的尖锐铜片触点刺穿对应铜线上的绝缘层，与铜线紧密接触并卡紧，这样就完成了 RJ45 连接器与双绞线的连接。

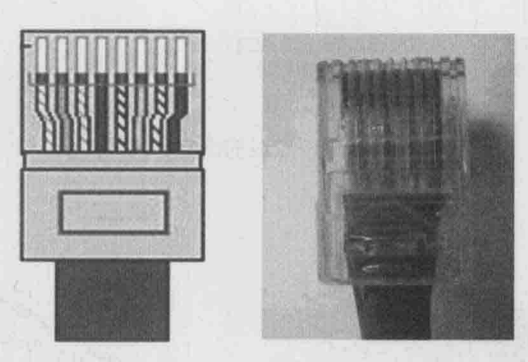


图 1-14 RJ45 连接器

3. 光纤

我们平时所说的光网络（或光传输网络、光通信网络），是指以光导纤维（简称光纤）作为传输介质的网络通信系统。这里的光导纤维，其实是一种玻璃纤维。在光网络通信系统中，光纤中传递的是一种波长在红外波段的、肉眼不可见的红外光。

光纤外面加上若干保护层后，便是我们通常所说的光缆（Optical Fiber Cable）。一条光缆中可以包含一根光纤，也可以包含多根光纤。图 1-15 示意了光纤/光缆的基本结构和实物外观。注意，外套、加强材料、缓冲层等都只是光纤的保护层，真正的光纤（光导纤维）指的是纤芯和覆层。纤芯和覆层的材质均是玻璃，所不同的是覆层玻璃体的折射系数约小于纤芯玻璃体的折射系数。

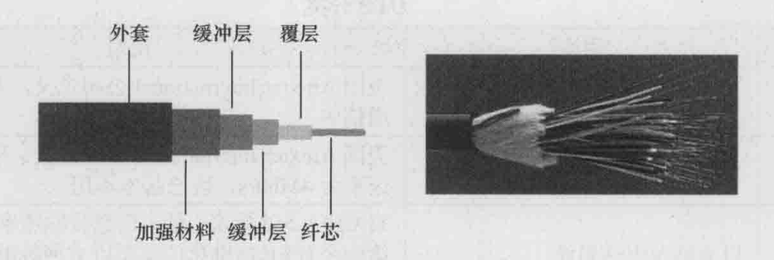


图 1-15 光纤/光缆的基本结构和实物外观

如图 1-16 所示，根据组成结构的差异，光纤可分为单模光纤和多模光纤。单模光纤的纤芯较细，覆层较厚；多模光纤的纤芯较粗，覆层较薄。光纤的粗细指的是覆层外围圆周的直径，大约为 125μm。人的头发直径为 17～181μm，亚洲人的头发直径大约为 120μm，所以一根光纤的粗细大致与亚洲人的一根头发相当。

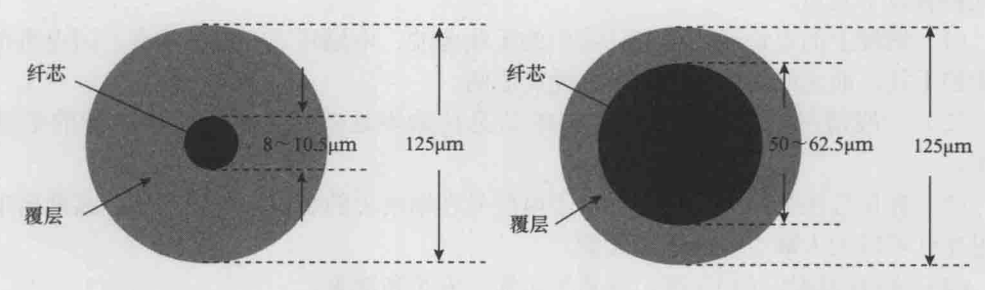


图 1-16　单模光纤与多模光纤

在单模光纤中，光是以单模方式进行传播的；而在多模光纤中，光是以多模方式进行传播的。关于单模传播方式与多模传播方式的比较分析，已经超出了本书所关注的知识范围，所以这里不做介绍。读者只需要知道以下几点。

（1）相比于单模光纤，多模光纤的纤芯较粗，生产工艺要求较为简单，所以生产成本较低，价格较便宜。

（2）多模光纤中的多模传播方式会引起模间色散，而模间色散会大大降低光信号的传输质量。模间色散会引起“脉冲展宽”效应，从而使得所传输的光信号产生畸变。单模光纤中不存在模间色散现象。

（3）传输距离越远，模间色散对光信号传输质量的影响越严重（光信号畸变程度越严重）。

（4）在传输距离相同的条件下，单模光纤比多模光纤能够支持更高的信息传输率；在信息传输率相同的条件下，单模光纤比多模光纤能够支持更大的传输距离。

（5）多模光纤多用于局域网络，传输距离较小（一般在几公里之内）；单模光纤多用于广域网，传输距离较大（可长达上千公里）。限制多模光纤传输距离的主要原因是减小模间色散对光信号传输质量的影响（弱化信号畸变程度）。

如同双绞线两端需要安装连接器一样，光缆的两端也需要安装光纤连接器。常见的光纤连接器有 ST 连接器、FC 连接器、SC 连接器、LC 连接器等，如图 1-17 所示。



图 1-17　光纤连接器

随着光网络通信系统的迅猛发展，光纤的使用也日益普及，并且在越来越多的场合替代了铜线的角色，这也就是所谓的“光进铜退”的趋势。与铜线相比，光纤带来的主

要优势有以下几点。

（1）铜线上的电信号会受到环境中的无线电波、电磁噪声、电磁感应、闪电雷击等因素的干扰，而光纤中的光信号不会受此影响。

（2）一般情况下，光纤能够支持的信息传输率远远高于铜线能够支持的信息传输率。

（3）光信号在光纤中的衰减远小于电信号在铜线上的衰减，所以在远距离通信中，使用光纤可以大大减少中继器的数量。

（4）光纤较铜线又轻又细，更易于运输、安装和部署。

1.4.2 通信方式

1. 串行通信与并行通信

串行通信是指在一条数据通道上，将数据一位一位（一比特一比特）地依次传输的通信方式。串行通信一次只能传输一个“0”或一个“1”。RS-232 线路上的通信方式就是一种串行通信方式。

并行通信是指在一组数据通道上，将数据一组一组地依次传输的通信方式。并行通信一次能够传输多个“0”和“1”。并行通信中，每一条数据通道上的传输原理都与串行通信类似。通常，并行通信是以字节为单位来进行传输的。计算机与数字投影仪之间的通信方式就是一种并行通信方式。

并行通信虽然可以大幅提升传输速率，但是也存在一些问题。例如，并行通信需要更多的数据通道，也就是需要更多的铜线或光纤，这无疑会增加网络的建设成本。另外，并行通信中，各数据通道上的信号同步要求非常苛刻。我们可以看一个例子，如图 1-18 所示，PC1 通过并行通信方式向 PC2 发送了两组数据。由于干扰或者别的什么原因，导致了数据 1 的第一位“1”比数据 1 的其他 7 位稍微晚了一点到达 PC2，于是 PC2 便会认为这一位已经丢失。然后，数据 1 的第一位“1”与数据 2 的第二位至第八位的到达时间几乎一致，于是 PC2 就会将数据 1 的第一位“1”当成是数据 2 的第一位，这样就产生了严重的误码情况。

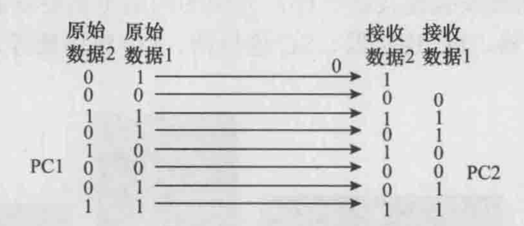


图 1-18 并行通信因同步问题而发生的误码现象

总之，并行通信中，各数据通道上的信号同步要求非常苛刻，并且信号传输距离越远，实现各数据通道上的信号同步就越困难，因此并行通信一般不适合远距离通信场合。

2. 单工、半双工、全双工通信方式

如图 1-19 所示，假定通信的双方分别为 A 和 B，则根据通信的指向性的不同，通信可以分为单工（Simplex）通信方式、半双工（Half-duplex）通信方式和全双工（Full-duplex）通信方式。

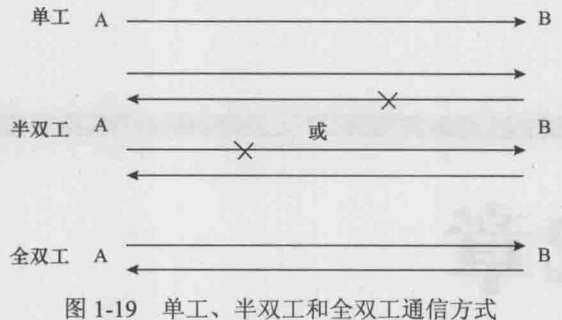


图1-19　单工、半双工和全双工通信方式

单工方式中，信息的流向只能由一方指向另一方。在图1-19所示的单工通信方式中，信息只能从A流向B，而不能从B流向A。也就是说，A只能向B发送数据，而B只能接收来自A的数据。广播通信系统、传统的模拟电视系统等都是单工通信方式的例子。

半双工方式中，信息的流向可以从A到B，也可以从B到A，但信息不能同时在两个方向上进行传递。也就是说，当A发送数据时，B只能接收数据；当B发送数据时，A只能接收数据。如果A和B同时发送数据，则通信双方都不能成功接收到对方发送的数据。对讲机系统就是半双工通信方式的例子。

全双工方式中，信息可以同时在两个方向上进行传递。也就是说，A、B 双方可以同时发送并接收数据。当A发送数据时，可以接收B正在发送的数据，反之亦然。我们平时所使用的固定电话通信系统和移动电话通信系统，都是全双工通信方式的例子。

1.4.3　练习题

1. (多选)在以太网环境中，以下哪些介质可以支持100Mbit/s的信息传输率？(　)

   A. 三类UTP　　B. 五类UTP　　C. 超五类UTP

2. （单选）在以太网环境中，规定三类、五类、超五类UTP的最大传输距离是多少？（　）

   A. 50m　　B. 100m　　C. 150m　　D. 200m

3. （单选）在传输距离相同的条件下，目前哪种介质所支持的信息传输速率最大？（　）

   A. 同轴电缆　　B. 光纤　　C. 双绞线

4. （单选）在信息传输速率相同的条件下，多模光纤的传输距离要比单模光纤小得多，主要原因是什么？（　）

   A. 多模光纤比单模光纤便宜一些

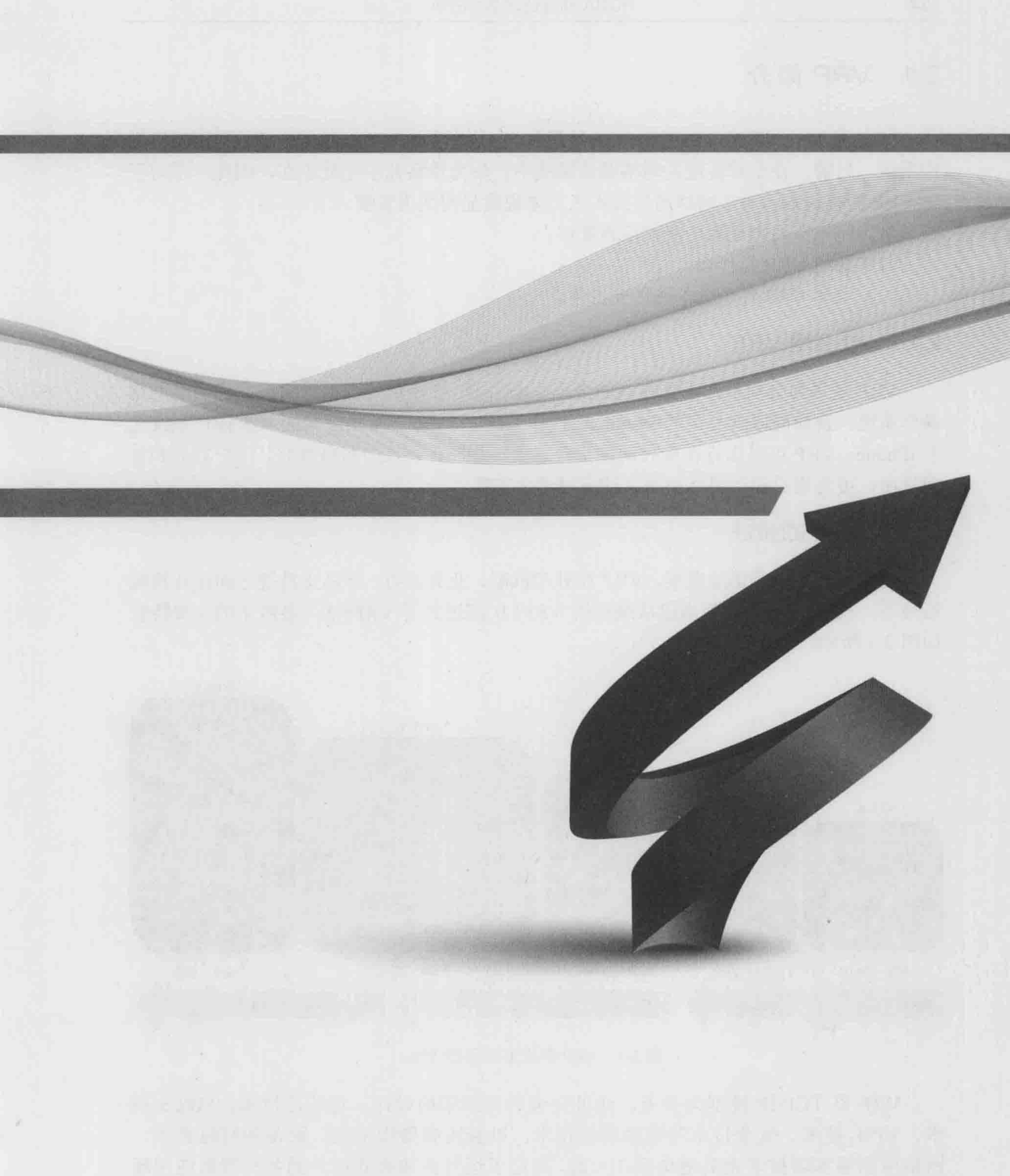
   B. 多模光纤中的模间色散现象会引起所传输的光信号产生畸变，且传输距离越大，畸变程度越严重

   C. 传输距离越大，多模光纤中的光信号的衰减越严重

5. （单选）在全球定位系统（GPS）中，卫星与地面接收终端之间的通信方式是哪种方式？（　）

   A. 单工通信方式　　B. 半双工通信方式　　C. 全双工通信方式

第2章 VRP基础



2.1 VRP 简介

VRP 是 Versatile Routing Platform 的简称，它是华为公司数据通信产品的通用网络操作系统。目前，在全球各地的网络通信系统中，华为设备几乎无处不在，因此，学习了解 VRP 的相关知识对于网络通信技术人员来说就显得尤为重要。

学习完本节内容之后，我们应该能够：

（1）知道 VRP 是什么；

（2）了解 VRP 各个版本的主要特性。

2.1.1 什么是 VRP

VRP 是华为公司从低端到高端的全系列路由器、交换机等数据通信产品的通用网络操作系统，就如同微软公司的 Windows 操作系统之于 PC，苹果公司的 iOS 操作系统之于 iPhone。VRP 可以运行在多种硬件平台之上，并拥有一致的网络界面、用户界面和管理界面，可为用户提供灵活而丰富的应用解决方案。

2.1.2 VRP 的演进

随着网络技术的迅速发展，VRP 在处理机制、业务能力、产品支持等方面也在持续地演进。目前，VRP 的版本已从最初的 VRP1.0 演进到了 VRP8.X，各版本的主要特性如图 2-1 所示。

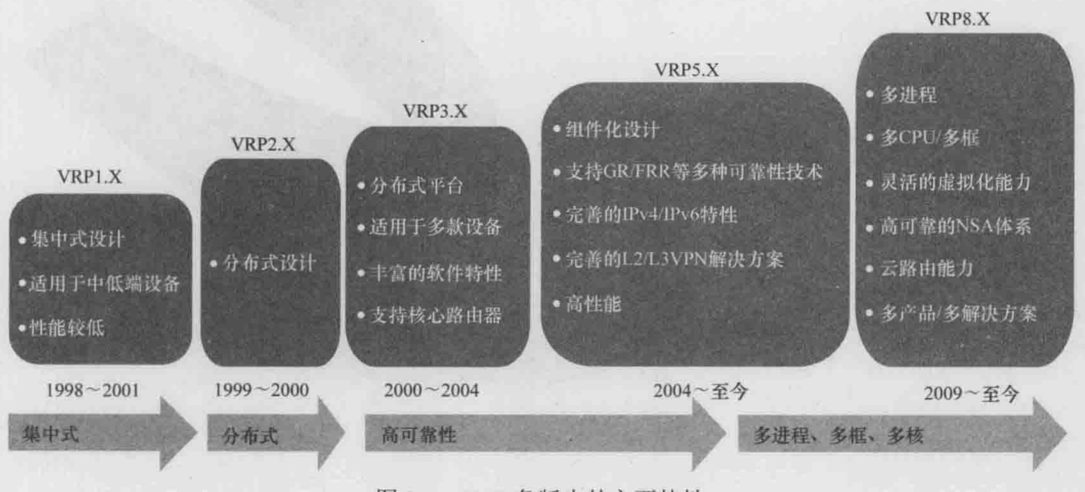


图 2-1 VRP 各版本的主要特性

VRP 以 TCP/IP 模型为参考，通过完善的体系架构设计，将路由技术、MPLS 技术、VPN 技术、安全技术等数据通信技术，以及实时操作系统、设备和网络管理、网络应用等多项技术完美地集成在一起，满足了运营商和企业用户的各种网络应用场景的需求。

目前，华为大部分适用于企业网络场景的中低端网络设备都是基于 VRP 5.X 的，本书后续各章节所涉及的 VRP 功能特性和配置描述均依据 VRP5.12。

2.2 VRP 命令行

要想实际操作华为网络设备，必须首先学会 VRP 命令行的使用方法。学习完本节内容之后，我们应该能够：

（1）了解命令行的概念、作用和其基本结构；

（2）理解用户视图、系统视图、接口视图之间的差异；

（3）熟悉命令级别和用户权限级别的划分情况；

（4）比较熟练地使用命令行。

2.2.1 命令行的基本概念

1. 命令行

华为网络设备功能的配置和业务的部署是通过 VRP 命令行来完成的。命令行是在设备内部注册的、具有一定格式和功能的字符串。一条命令行由关键字和参数组成，关键字是一组与命令行功能相关的单词或词组，通过关键字可以唯一确定一条命令行，本书正文中采用**加粗字体**方式来标识命令行的关键字。参数是为了完善命令行的格式或指示命令的作用对象而指定的相关单词或数字等，包括整数、字符串、枚举值等数据类型，本书正文中采用斜体字体方式来标识命令行的参数。例如，测试设备间连通性的命令行 **ping** *ip-address* 中，**ping** 为命令行的关键字，*ip-address* 为参数（取值为一个 IP 地址）。

新购买的华为网络设备，初始配置为空。若希望它能够具有诸如文件传输、网络互通等功能，则需要首先进入到该设备的命令行界面，并使用相应的命令进行配置。

2. 命令行界面

命令行界面是用户与设备之间的文本类指令交互的界面，就如同 Windows 操作系统中的 DOS（Disk Operation System）窗口一样。VRP 命令行界面如图 2-2 所示。

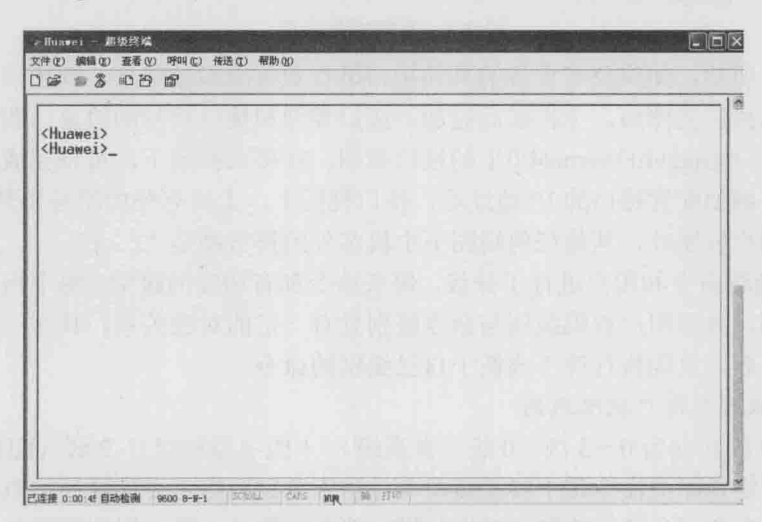


图 2-2 VRP 命令行界面/用户视图界面

VRP 命令的总数达数千条之多，为了实现对它们的分级管理，VRP 系统将这些命令

按照功能类型的不同分别注册在了不同的视图之下。

3. 命令行视图

命令行界面分成了若干种命令行视图，使用某个命令行时，需要先进入到该命令行所在的视图。最常用的命令行视图有用户视图、系统视图和接口视图，三者之间既有联系，又有一定的区别。

如图 2-2 所示，进入命令行界面后，首先进入的就是用户视图。提示符“<Huawei>”中，“< >”表示是用户视图，“Huawei”是设备缺省的主机名。在用户视图下，用户可以了解设备的基础信息、查询设备状态，但不能进行与业务功能相关的配置。如果需要对设备进行业务功能配置，则需要进入到系统视图。

如图 2-3 所示，在用户视图下使用 **system-view** 命令，便可以进入到系统视图，此时的提示符中使用了方括号“[ ]”。系统视图下可以使用绝大部分的基础功能配置命令。另外，系统视图还提供了进入其他视图的入口；若希望进入其他视图，必须先进入到系统视图。

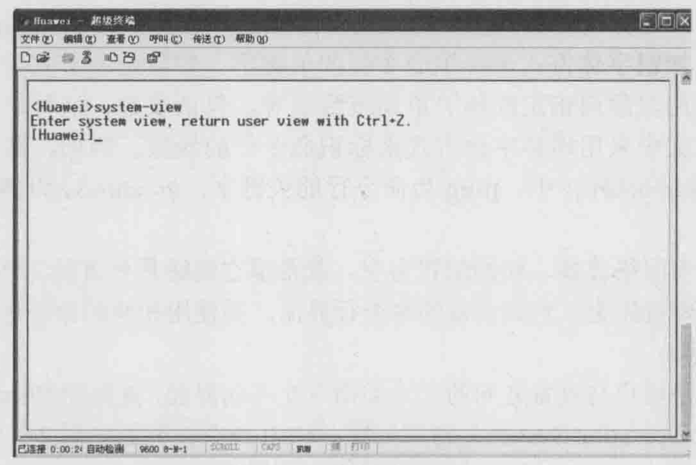


图 2-3 系统视图界面

如图 2-4 所示，如果要对设备的具体接口进行业务或参数配置，则还需要进入到接口视图。进入接口视图后，主机名后追加了接口类型和接口编号的信息。图 2-4 显示的是如何进入接口 GigabitEthernet4/0/1 的接口视图。在接口视图下，可以完成对相应接口的配置操作，例如配置接口的 IP 地址等。接口视图下，主机名外的符号仍然是“[ ]”。事实上，除用户视图外，其他任何视图下主机名外的符号都是“[ ]”。

VRP 系统将命令和用户进行了分级，每条命令都有相应的级别，每个用户也都有自己的权限级别，并且用户权限级别与命令级别具有一定的对应关系。具有一定权限级别的用户登录以后，只能执行等于或低于自己级别的命令。

4. 命令级别与用户权限级别

VRP 命令级别分为 0～3 级：0 级（参观级）、1 级（监控级）、2 级（配置级）、3 级（管理级）。网络诊断类命令属于参观级命令，用于测试网络是否连通等。监控级命令用于查看网络状态和设备基本信息。对设备进行业务配置时，需要用到配置级命令。对于一些特殊的功能，如上传或下载配置文件，则需要用到管理级命令。



图 2-4 接口视图界面

用户权限分为 0～15 共 16 个级别。默认情况下，3 级用户就可以操作 VRP 系统的所有命令，也就是说 4～15 级的用户权限在默认情况下是与 3 级用户权限一致的。4～15 级的用户权限一般与提升命令级别的功能一起使用，例如当设备管理员较多时，需要在管理员中再进行权限细分，这时可以将某条关键命令所对应的用户级别提高，如提高到 15 级，这样一来，缺省的 3 级管理员便不能再使用该关键命令。

命令级别与用户权限级别的对应关系如表 2-1 所示。

表 2-1 用户权限级别与命令级别的对应关系

用户级别	命令级别	说明
0	0	网络诊断类命令（**ping**，**tracert**）、从本设备访问其他设备的命令（**telnet**）等
1	0、1	系统维护命令，包括 **display** 等。但并不是所有的 **display** 命令都是监控级的，例如 **display current-configuration** 和 **display saved-configuration** 都是管理级命令
2	0、1、2	业务配置命令，包括路由、各个网络层次的命令等
3～15	0、1、2、3	涉及系统基本运行的命令，如文件系统、FTP 下载、配置文件切换命令、用户管理命令、命令级别设置命令、系统内部参数设置命令等，还包括故障诊断的 **debugging** 命令

 注意

建议不要随意修改缺省的命令级别。如果确实需要修改，则应该在专业人员的指导下进行修改，以免造成操作和维护上的不便，甚至给设备带来安全隐患。

2.2.2 命令行的使用方法

1. 进入命令视图

用户进入 VRP 系统后，首先进入的就是用户视图。如果出现<Huawei>，并有光标在“>”右边闪动，则表明用户已成功进入了用户视图。

```
<Huawei>
```

进入用户视图后，便可以通过命令来了解设备的基础信息、查询设备状态等。如果需要对 GigabitEthernet1/0/0 接口进行配置，则需先使用 **system-view** 命令进入系统视图，再使用 **interface** *interface-type interface-number* 命令进入相应的接口视图。

```
<Huawei> system-view
[Huawei]                //已进入系统视图
[Huawei] interface gigabitethernet 1/0/0
[Huawei-GigabitEthernet1/0/0]          //已进入接口视图
```

2. 退出命令视图

quit 命令的功能是从任何一个视图退出到上一层视图。例如，接口视图是从系统视图进入的，所以系统视图是接口视图的上一层视图。

```
[Huawei-GigabitEthernet1/0/0] quit
[Huawei]                //已退出到系统视图
```

如果希望继续退出至用户视图，可再次执行 **quit** 命令。

```
[Huawei] quit
<Huawei>                //已退出到用户视图
```

有些命令视图的层级很深，从当前视图退出到用户视图，需要多次执行 **quit** 命令。使用 **return** 命令，可以直接从当前视图退出到用户视图。

```
[Huawei-GigabitEthernet1/0/0] return
<Huawei>                //已退出到用户视图
```

另外，在任意视图下，使用快捷键<Ctrl+Z>，可以达到与使用 **return** 命令相同的效果。

3. 输入命令行

VRP 系统提供了丰富的命令行输入方法，支持多行输入，每条命令最大长度为 510 个字符，命令关键字不区分大小写，同时支持不完整关键字输入。表 2-2 列出了命令行输入过程中常用的一些功能键。

表 2-2 VRP 命令行编辑功能键

功能键	功能
退格键 BackSpace	删除光标位置的前一个字符，光标左移；若已经到达命令起始位置，则停止
左光标键←或<Ctrl+B>	光标向左移动一个字符位置；若已经到达命令起始位置，则停止
右光标键→或<Ctrl+F>	光标向右移动一个字符位置；若已经到达命令尾部，则停止
删除键 Delete	删除光标所在位置的一个字符，光标位置保持不动，光标后方字符向左移动一个字符位置；若已经到达命令尾部，则停止
上光标键↑或<Ctrl+P>	显示上一条历史命令。如果需显示更早的历史命令，可以重复使用该功能键
下光标键↓或<Ctrl+N>	显示下一条历史命令，可重复使用该功能键

4. 不完整关键字输入

为了提高命令行输入的效率和准确性，VRP 系统能够支持不完整的关键字输入功能，即在当前视图下，当输入的字符能够匹配唯一的关键字时，可以不必输入完整的关键字。例如，当需要输入命令 **display current-configuration** 时，可以通过输入 **d cu**、**di cu** 或 **dis cu** 来实现，但不能输入 **d c** 或 **dis c** 等，因为系统内有多条以 **d c**、**dis c** 开头的命令，如：**display cpu-defend**、**display clock** 和 **display current-configuration**。

5. 在线帮助

在线帮助是 VRP 系统提供的一种实时帮助功能。在命令行输入过程中，用户可以随

时键入“？”以获得在线帮助信息。命令行在线帮助可分为完全帮助和部分帮助。

关于完全帮助，我们来看一个例子。假如我们希望查看设备的当前配置情况，但在进入用户视图后不知道下一步该如何操作，这时就可以键入“？”，得到如下的回显帮助信息。

```
<Huawei> ?
User view commands:
  arp-ping                  ARP-ping
  autosave                  <Group> autosave command group
  backup                    Backup  information
  cd                        Change current directory
  clear                     Clear
  clock                     Specify the system clock
  cls                       Clear screen
  compare                   Compare configuration file
  copy                      Copy from one file to another
  debugging                 <Group> debugging command group
  delete                    Delete a file
  dialer                    Dialer
  dir                       List files on a filesystem
  display                   Display information
  factory-configuration     Factory configuration
  fixdisk                   Try to restory disk
---- More ----
```

从显示的关键字中可以看到“**display**”，对此关键字的解释为Display information。我们自然会想到，要查看设备的当前配置情况，很可能会用到“**display**”这个关键字。于是，按任意字母键退出帮助后，键入**display**和空格，再键入问号“？”，得到如下的回显帮助信息。

```
<Huawei> display ?
  accounting-scheme         Accounting scheme
  acl                       <Group> acl command group
  actual                    Current actual
  ap                        <Group> ap command group
  bfd                       Specify BFD（Bidirectional Forwarding Detection）
  bgp                       BGP information
  binding                   Display binding relation of profile
  bridge-link               Bridge link
  bridge-profile            Display Bridge profile
  bridge-whitelist          Bridge Whitelist
  bssid-decode              Display bssid detail information
  calibrate                 Global calibrate
  clock                     Clock status and configuration information
  config                    System config
  cpu-defend                Configure CPU defend policy
  cpu-usage                 Cpu usage information
  current-configuration     Current configuration
---- More ----
```

从回显信息中，我们发现了“**current-configuration**”。通过简单的分析和推理，我们便知道，要查看设备的当前配置情况，应该输入的命令行是“**display current-configuration**”。

我们再来看一个部分帮助的例子。通常情况下，我们不会完全不知道整个需要输入的命令行，而是知道命令行关键字的部分字母。假如我们希望输入**display current-configuration**命令，但不记得完整的命令格式，只是记得关键字**display**的开头

字母为 **dis**，**current-configuration** 的开头字母为 **c**。此时，我们就可以利用部分帮助功能来确定出完整的命令。键入 **dis** 后，再键入问号“？”。

```
<Huawei> dis ?
  display   Display information
```

回显信息表明，以 **dis** 开头的关键字只有 **display**。根据不完整关键字输入原则，用 **dis** 就可以唯一确定关键字 **display**。所以，在输入 **dis** 后直接输入空格，然后输入 **c**，最后输入“？”，以获取下一个关键字的帮助。

```
<Huawei> dis c ?
  Cellular                    Cellular interface
  calibrate                   Global calibrate
  capwap                      CAPWAP
  channel                     Informational channel status and configuration
                              information
  clock                       Clock status and configuration information
  config                      System config
  controller                  Specify controller
  cpos                        CPOS controller
  cpu-defend                  Configure CPU defend policy
  cpu-usage                   Cpu usage information
  current-configuration       Current configuration
  cwmp                        CPE WAN Management Protocol
```

回显信息表明，关键字 **display** 后，以 **c** 开头的关键只有为数不多的十几个，从中很容易找到 **current-configuration**。至此，我们便从 **dis** 和 **c** 这样的记忆片段中恢复出了完整的命令行 **display current-configuration**。

6. 快捷键

快捷键的使用可以进一步提高命令行的输入效率。VRP 系统已经定义了一些快捷键，称为系统快捷键。系统快捷键功能固定，用户不能再重新定义。常见的系统快捷键如表 2-3 所示。

表 2-3　　常见的 VRP 系统快捷键

快捷键	功能
CTRL_A	将光标移动到当前行的开始
CTRL_E	将光标移动到当前行的末尾
ESC_N	将光标向下移动一行
ESC_P	将光标向上移动一行
CTRL_C	停止当前正在执行的功能
CTRL_Z	返回到用户视图，功能相当于 return 命令
<Tab>键	部分帮助的功能，输入不完整的关键字后按下<Tab>键，系统自动补全关键字

VRP 系统还允许用户来自定义一些快捷键，但自定义快捷键可能会与某些操作命令发生混淆，所以一般情况下最好不要自定义快捷键。

2.3 登录设备

学习完本节内容之后，我们应该能够：

（1）通过 Console 口登录设备；

（2）通过 MiniUSB 口登录设备；

（3）在 PC 端安装 MiniUSB 口的驱动程序。

2.3.1 通过 Console 口登录设备

图 2-5 展示了 Console 通信电缆的实物外观及组成结构。D 型连接器用来连接计算机的串口；网口连接器就是 RJ-45 水晶头，用来连接网络设备的 Console 接口。

下面我们以使用 Windows XP 系统的超级终端软件为例，讲解 PC 机如何通过网络设备的 Console 口登录到网络设备。

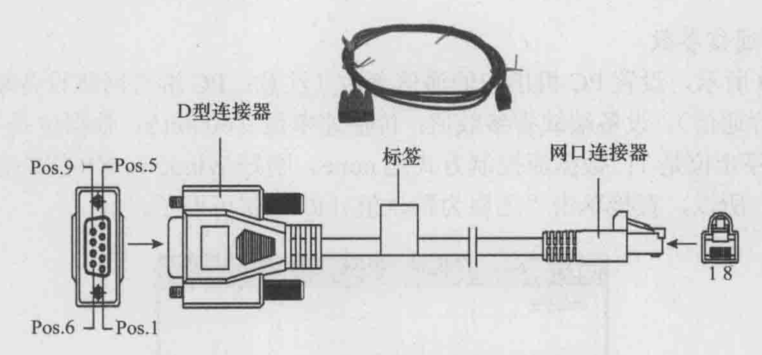


图 2-5 Console 通信电缆实物外观及组成结构

1. 线缆连接

如图 2-6 所示，将 Console 通信电缆的 D 型连接器插入 PC 机的串口插座，再将 RJ-45 水晶头插入设备的 Console 口中。

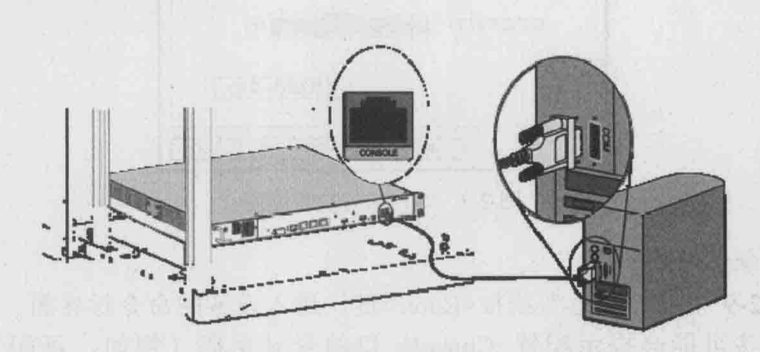


图 2-6 用 Console 通信电缆连接 PC 机和网络设备

2. 新建连接并设置连接端口

对 PC 机和设备进行上电。在 PC 上单击“开始→所有程序→附件→通信→超级终端”打开超级终端软件，并新建一个连接，如图 2-7 所示。然后，如图 2-8 所示，设置连接端口。因为 PC 机可能会存在多个串口，这里选择的应该是连接 Console 电缆的那个串口，假设为 COM1。

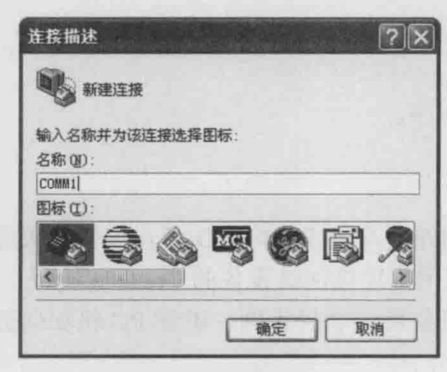


图 2-7　新建连接

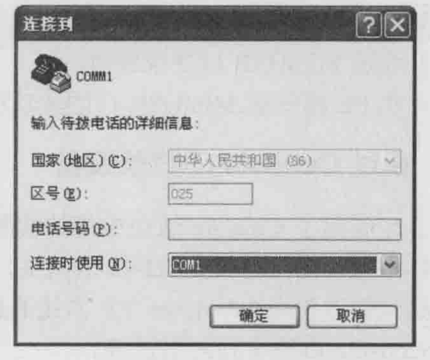


图 2-8　设置连接端口

3. 设置通信参数

如图 2-9 所示，设置 PC 机串口的通信参数（注意：PC 端与网络设备端的参数保持一致才能进行通信）。设备端缺省参数值：传输速率是 9 600bit/s；数据位是 8；奇偶校验位是 none；停止位是 1；数据流控制方式是 none。刚好 Windows XP 的缺省配置就是设备的缺省值，所以，直接单击“还原为默认值（R）”就可以了。



图 2-9　端口通信参数设置

4. 进入命令行界面

单击图 2-9 中的“确定”或按<Enter>键，进入设备的命令行界面。首次登录新设备时，设备可能会提示配置 Console 口的登录密码（例如，可配置密码为：huawei2013）。配置登录密码后，设备还会提示是否关闭 Auto-Config 功能，此功能可以实现设备自动配置。如果打算使用命令行配置设备时，应输入“y”关闭此功能，否则输入“n”。

```
Please configure the login password （maximum length 16）: huawei2013
<Huawei>
 Warning: Auto-Config is working. Before configuring the device, stop Auto-Config. If you perform configurations when Auto-Config is running, the DHCP, routing, DNS, and VTY configurations will be lost. Do you want to stop Auto-Config ? [y/n]: y
<Huawei>
```

完成以上步骤后，设备显示<Huawei>，表示已进入用户视图，接下来就可以对设备

进行基本配置了。

通过 Console 口登录设备时需要用户的 PC 机上有串口。如果 PC 机上没有串口，则可以通过 MiniUSB 口登录设备。

2.3.2　通过 MiniUSB 口登录设备

图 2-10 展示了 MiniUSB 线缆的实物外观。MiniUSB 线缆的一端是 USB 接口，用来连接 PC 机的 USB 接口；另一端是 MiniUSB 接口，用来连接网络设备的 MiniUSB 接口。

下面我们以使用 Windows XP 系统的超级终端软件为例，讲解 PC 机如何通过网络设备的 MiniUSB 口登录到网络设备。

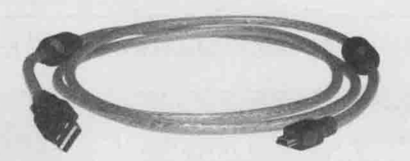


图 2-10　MiniUSB 线缆实物外观

1. 线缆连接

如图 2-11 所示，用 MiniUSB 线缆将 PC 机的 USB 口和设备的 MiniUSB 口相连。

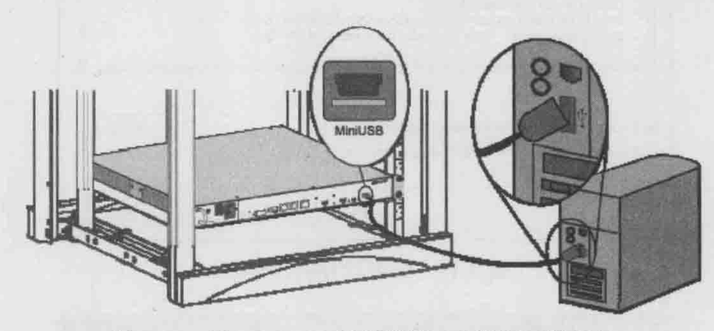


图 2-11　用 MiniUSB 线缆连接 PC 机和网络设备

2. MiniUSB 驱动程序安装

通过 MiniUSB 接口登录设备，需要在 PC 端安装 MiniUSB 驱动程序，步骤如下。

（1）MiniUSB 的驱动程序 AR_MiniUSB_driver 可以从华为公司企业业务支持网站（http：//support.huawei.com/enterprise）获取。目前，该驱动程序仅支持 Windows XP/VISTA/7 操作系统。驱动程序下载成功之后，在 PC 端双击驱动程序的安装文件并单击“Next”，如图 2-12 所示。

（2）选择“I accept the terms in the license agreement”并单击“Next”，如图 2-13 所示。

（3）单击“Change”可以更改驱动程序解压的路径。如果不需要更改驱动程序解压的路径，可以直接单击“Next”，如图 2-14 所示。



图 2-12　PC 端运行 MiniUSB 驱动安装程序

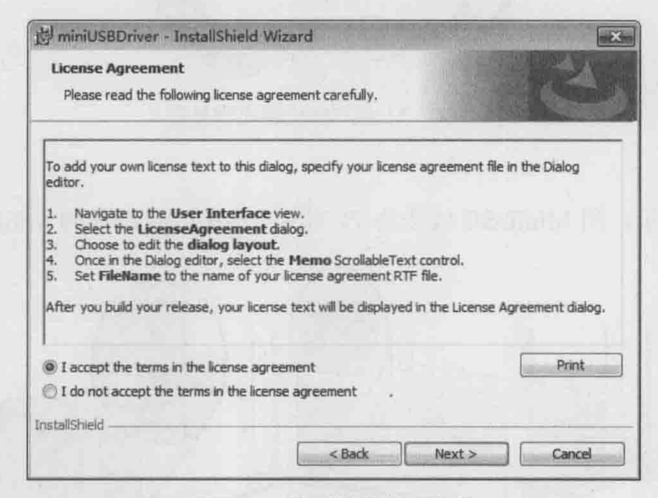


图 2-13　接受软件协议条款

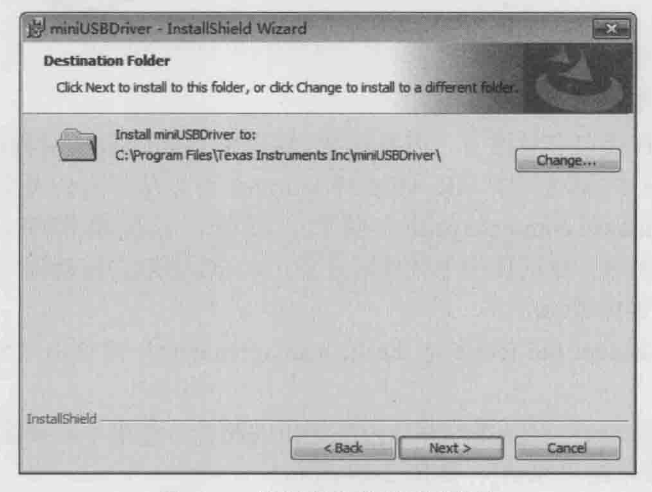


图 2-14　选择驱动程序解压路径

（4）单击“Install”后进行解压，完成后单击“Finish”结束解压，如图 2-15 所示。



图 2-15　完成驱动程序的解压

（5）在（3）指定的解压路径下找到“DISK1”文件夹，双击“setup.exe”，如图 2-16 所示。

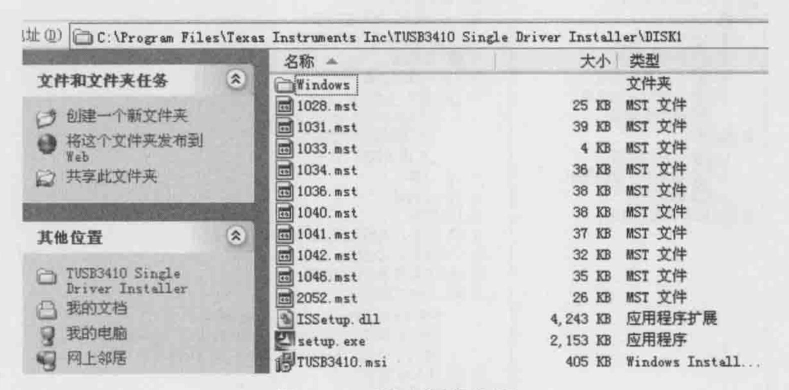


图 2-16　驱动程序位置

（6）单击“下一步”，选择“我接受许可证协议中的条款（A）”，并单击“下一步”进入驱动安装，然后单击“完成”，结束驱动程序的安装，如图 2-17 所示。

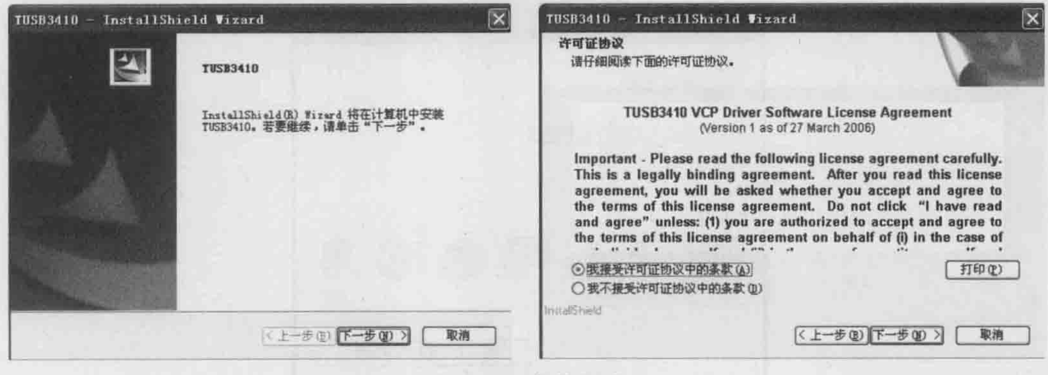


图 2-17　安装驱动



图 2-17 安装驱动（续）

（7）驱动程序安装完成后，鼠标右键单击“我的电脑”，单击“管理->设备管理器->端口（COM 和 LPT）”，显示的“TUSB3410 Device”即为已安装的设备，如图 2-18 所示。

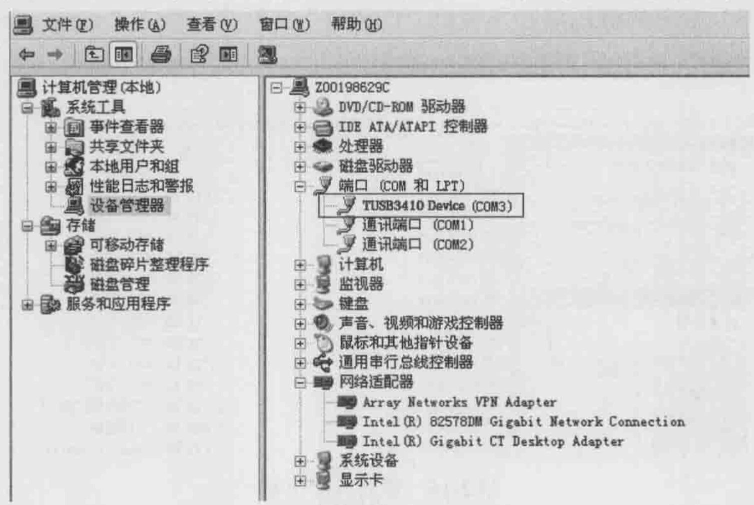


图 2-18 查看设备管理器

3. 新建连接

在 PC 机上打开 Windows XP 自带的超级终端软件，创建一个新的连接，如图 2-19 所示。

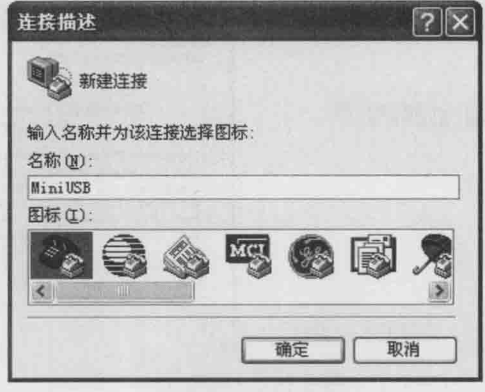


图 2-19 新建连接

4. 设置连接端口

设置连接端口为步骤（7）所显示的 COM 口，如图 2-20 所示。

5. 设置通信参数

MiniUSB 口和 Console 口使用的设备缺省参数值都一样，所以此处可以直接单击"还原为默认值（R）"，如图 2-21 所示。

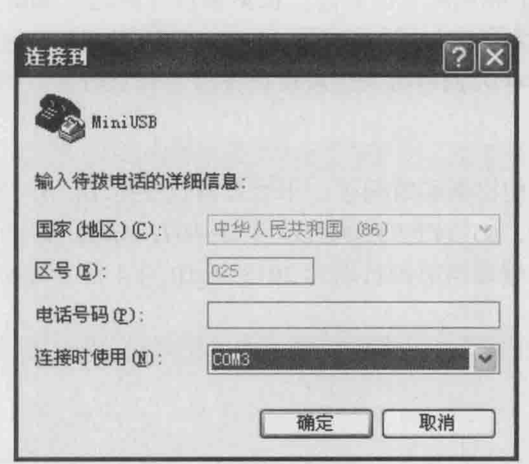


图 2-20 设置连接端口

图 2-21 端口通信参数设置

6. 进入命令行界面

单击图 2-21 中的"确定"，或按<Enter>键，进入设备的命令行界面。

2.4 基本配置

学习完本节内容之后，我们应该能够：

（1）配置设备名称；

（2）配置设备系统时间；

（3）配置设备 IP 地址；

（4）完成用户界面的基本配置。

2.4.1 配置设备名称

命令行界面中的尖括号"< >"或方括号"[ ]"中包含有设备的名称，也称为设备主机名。缺省情况下，设备名称为"Huawei"。为了更好地区分不同的设备，通常需要修改设备名称。我们可以通过命令 **sysname** *host-name* 来对设备名称进行修改，其中 **sysname** 是命令行的关键字，*host-name* 为参数，表示希望设置的设备名称。

例如，通过如下操作，就可以将设备名称设置为 Huawei-AR-01。

```
<Huawei> system-view
Enter system view, return user view with Ctrl+Z
[Huawei] sysname Huawei-AR-01
[Huawei-AR-01]
```

2.4.2 配置设备系统时钟

华为设备出厂时默认采用了协调世界时（UTC），但没有配置时区，所以在配置设备系统时钟前，需要了解设备所在的时区。

设置时区的命令行为 **clock timezone** *time-zone-name* {**add**|**minus**}*offset*，其中 *time-zone-name* 为用户定义的时区名，用于标识配置的时区；根据偏移方向选择 **add** 和 **minus**，正向偏移（UTC 时间加上偏移量为当地时间）选择 **add**，负向偏移（UTC 时间减去偏移量为当地时间）选择 **minus**；*offset* 为偏移时间。假设设备位于北京时区，则相应的配置应该是：

```
[Huawei-AR-01] clock timezone BJ add 08:00
```

设置好时区后，就可以设置设备当前的日期和时间了。华为设备仅支持 24 小时制，使用的命令行为 **clock datetime** *HH：MM：SS YYYY-MM-DD*，其中 *HH：MM：SS* 为设置的时间，*YYYY-MM-DD* 为设置的日期。假设当前的日期为 2013 年 10 月 4 日，时间是凌晨 02：06：00，则相应的配置应该是：

```
[Huawei-AR-01] clock datetime 02:06:00 2013-10-04
```

2.4.3 配置设备 IP 地址

用户可以通过不同的方式登录到设备命令行界面，包括 Console 口登录、MiniUSB 口登录以及 Telnet 登录。首次登录新设备时，由于新设备为空配置设备，所以只能通过 Console 口或 MiniUSB 口登录。首次登录到新设备后，便可以给设备配置一个 IP 地址，然后开启 Telnet 功能。

IP 地址是针对设备接口的配置，通常一个接口配置一个 IP 地址。配置接口 IP 地址的命令为 **ip address** *ip-address*{*mask*|*mask-length*}，其中 **ip address** 是命令关键字，*ip-address* 为希望配置的 IP 地址。*mask* 表示点分十进制方式的子网掩码；*mask-length* 表示长度方式的子网掩码，即掩码中二进制数 1 的个数。

假设设备 Huawei-AR-01 的管理接口为 Ethernet 1/0/0，分配的 IP 地址为 10.1.1.100，子网掩码为 255.255.255.0，则相应的配置应该是：

```
<Huawei-AR-01> system-view
[Huawei-AR-01] interface ethernet 1/0/0          //进入接口视图
[Huawei-AR-01-Ethernet1/0/0] ip address 10.1.1.100 255.255.255.0
```

说明

该例中的子网掩码也可以用掩码的长度 24 直接表示。

2.4.4 用户界面配置

1. 用户界面的概念

用户在与设备进行信息交互的过程中，不同的用户拥有各自不同的用户界面。使用 Console 口登录设备的用户，其用户界面对应了设备的物理 Console 接口；使用 Telnet 登录设备的用户，其用户界面对应了设备的虚拟 VTY（Virtual Type Terminal）接口。

说明

不同设备支持的 VTY 总数可能不同。

如果希望对不同的用户进行登录控制，则需要首先进入到对应的用户界面视图进行相应的配置（如：规定用户权限级别、设置用户名和密码等）。例如，假设规定通过 Console 口登录的用户的权限级别为 2 级，则相应的操作如下。

```
<Huawei> system-view
[Huawei] user-interface console 0       //进入 Console 口用户的用户界面视图
[Huawei-ui-console0] user privilege level 2
```

如果有多个用户登录设备，因为每个用户都会有自己的用户界面，那么设备如何识别这些不同的用户界面呢？

2. 用户界面的编号

用户登录设备时，系统会根据该用户的登录方式，自动分配一个当前空闲且编号最小的相应类型的用户界面给该用户。用户界面的编号包括以下两种。

（1）相对编号

相对编号的形式是：用户界面类型+序号。一般地，一台设备只有 1 个 Console 口（插卡式设备可能有多个 Console 口，每个主控板提供 1 个 Console 口），VTY 类型的用户界面一般有 15 个（缺省情况下，开启了其中的 5 个）。所以，相对编号的具体呈现如下。

- Console 口的编号：CON 0。

- VTY 的编号：第一个为 VTY 0，第二个为 VTY 1，以此类推。

（2）绝对编号

绝对编号仅仅是一个数值，用来唯一标识一个用户界面。绝对编号与相对编号具有一一对应的关系：Console 用户界面的相对编号为 CON 0，对应的绝对编号为 0；VTY 用户界面的相对编号为 VTY 0～VTY 14，对应的绝对编号为 129～143。

使用 **display user-interface** 命令可以查看设备当前支持的用户界面信息，操作如下。

```
<Huawei> display user-interface
  Idx      Type      Tx/Rx     Modem  Privi  ActualPrivi  Auth  Int
  0        CON 0     9600      -      15     -            -
+ 129      VTY 0               -      15     15           P     -
  130      VTY 1               -      15     -            A     -
  131      VTY 2               -      15     -            A     -
  132      VTY 3               -      15     -            A     -
  133      VTY 4               -      15     -            P     -
  134      VTY 5               -      15     -            P     -
  135      VTY 6               -      15     -            P     -
  136      VTY 7               -      15     -            P     -
  137      VTY 8               -      15     -            P     -
  138      VTY 9               -      15     -            P     -
  139      VTY 10              -      15     -            P     -
  140      VTY 11              -      15     -            P     -
  141      VTY 12              -      15     -            P     -
  142      VTY 13              -      15     -            P     -
  143      VTY 14              -      15     -            P     -
UI（s） not in async mode -or- with no hardware support:
1-128
  +     : Current UI is active.
```

```
F      : Current UI is active and work in async mode.
Idx : Absolute index of UIs.
Type : Type and relative index of UIs.
Privi : The privilege of UIs.
ActualPrivi : The actual privilege of user-interface.
Auth : The authentication mode of UIs.
    A : Authenticate use AAA.
    N : Current UI need not authentication.
    P : Authenticate use current UI's password.
Int : The physical location of UIs.
```

回显信息中，第一列 Idx 表示绝对编号，第二列 Type 为对应的相对编号。

3. 用户验证

每个用户登录设备时都会有一个用户界面与之对应。那么，如何做到只有合法用户才能登录设备呢？答案是通过用户验证机制。设备支持的验证方式有 3 种：Password 验证、AAA 验证和 None 验证。

（1）Password 验证：只需输入密码，密码验证通过后，即可登录设备。缺省情况下，设备使用的是 Password 验证方式。使用该方式时，如果没有配置密码，则无法登录设备。

（2）AAA 验证：需要输入用户名和密码，只有输入正确的用户名和其对应的密码时，才能登录设备。由于需要同时验证用户名和密码，所以 AAA 验证方式的安全性比 Password 验证方式高，并且该方式可以区分不同的用户，用户之间互不干扰。所以，使用 Telnet 登录时，一般都采用 AAA 验证方式。

（3）None 验证：不需要输入用户名和密码，可直接登录设备，即无需进行任何验证。为安全起见，不推荐使用这种验证方式。

用户验证机制保证了用户登录的合法性。缺省情况下，通过 Telnet 登录的用户，在登录后的权限级别是 0 级。

4. 用户权限级别

2.2.1 小节已经对用户权限级别的含义以及它与命令级别的对应关系进行了描述。用户权限级别也称为用户级别，默认情况下，用户级别在 3 级及以上时，便可以操作设备的所有命令。某个用户的级别，可以在对应用户界面视图下执行 **user privilege level** *level* 命令进行配置，其中 *level* 为指定的用户级别。

有了以上这些关于用户界面的相关知识后，我们接下来通过两个实例来说明 VTY 和 Console 用户界面的配置方法。

实例 1：配置 VTY 用户界面。

VTY 用户界面对应于使用 Telnet 方式登录的用户。考虑到 Telnet 是远程登录，容易存在安全隐患，所以在用户验证方式上采用了 AAA 验证。一般地，设备调试阶段需要登录设备的人员较多，并且需要进行业务方面的配置，所以通常配置最大 VTY 用户界面数为 15，即允许最多 15 个用户同时使用 Telnet 方式登录到设备。同时，应将用户级别设置为 2 级，即配置级，以便可以进行正常的业务配置。

（1）配置最大 VTY 用户界面数为 15

配置最大 VTY 用户界面数使用的命令是 **user-interface maximum-vty** *number*。如果希望配置最大 VTY 用户界面数为 15 个，则 *number* 应取值为 15。

```
<Huawei> system-view
[Huawei] user-interface maximum-vty 15
```

（2）进入 VTY 用户界面视图

使用 **user-interface vty** *first-ui-number* [*last-ui-number*]命令进入 VTY 用户界面视图，其中 *first-ui-number* 和 *last-ui-number* 为 VTY 用户界面的相对编号，方括号“[ ]”表示该参数为可选参数。假设现在需要对 15 个 VTY 用户界面进行整体配置，则 *first-ui-number* 应取值为 0，*last-ui-number* 取值为 14。

```
[Huawei] user-interface vty 0 14
[Huawei-ui-vty0-14]              //进入了 VTY 用户界面视图
```

（3）配置 VTY 用户界面的用户级别为 2 级

配置用户级别的命令为 **user privilege level** *level*。因为现在需要配置用户级别为 2 级，所以 *level* 的取值为 2。

```
[Huawei-ui-vty0-14] user privilege level 2
```

（4）配置 VTY 用户界面的用户验证方式为 AAA

配置用户验证方式的命令为 **authentication-mode**{**aaa**|**none**|**password**}，其中大括号“{ }”表示其中的参数应任选其一。

```
[Huawei-ui-vty0-14] authentication-mode aaa
```

（5）配置 AAA 验证方式的用户名和密码

首先退出 VTY 用户界面视图，执行命令 **aaa**，进入 AAA 视图。再执行命令 **local-user** *user-name* **password cipher** *password*，配置用户名和密码。*user-name* 表示用户名，*password* 表示密码，关键字 **cipher** 表示配置的密码将以密文形式保存在配置文件中。最后，执行命令 **local-user** *user-name* **service-type telnet**，定义这些用户的接入类型为 Telnet。

```
[Huawei--ui-vty0-14] quit
[Huawei] aaa
[Huawei-aaa] local-user admin password cipher admin@123
[Huawei-aaa] local-user admin service-type telnet
[Huawei-aaa] quit
```

配置完成后，当用户通过 Telnet 方式登录设备时，设备会自动分配一个编号最小的可用 VTY 用户界面给用户使用，进入命令行界面之前需要输入上面配置的用户名（admin）和密码（admin@123）。

实例 2：配置 Console 用户界面。

Console 用户界面对应于从 Console 口直连登录的用户，一般采用 Password 验证方式。通过 Console 口登录的用户一般为网络管理员，需要最高级别的用户权限。

（1）进入 Console 用户界面

进入 Console 用户界面使用的命令为 **user-interface console** *interface-number*，*interface-number* 表示 Console 用户界面的相对编号，取值为 0。

```
[Huawei] user-interface console 0
```

（2）配置用户界面

在 Console 用户界面视图下配置验证方式为 Password 验证，并配置密码为 admin@123，且密码将以密文形式保存在配置文件中。

配置用户界面的用户验证方式的命令为 **authentication-mode**{**aaa**|**none**|**password**}。

使用 **set authentication password cipher** *password* 命令，配置密文密码。

```
[Huawei-ui-console0] authentication-mode password
[Huawei-ui-console0] set authentication password cipher admin@123
```

配置完成后，配置信息会保存在设备的内存中，使用命令 **display current-configuration** 即可进行查看。如果不进行存盘保存，则这些信息在设备通电或重启时将会丢失。

2.5 配置文件管理

学习完本节内容之后，我们应该能够：

（1）熟悉 3 个基本概念：当前配置、配置文件、下次启动的配置文件；

（2）完成设备当前配置的保存；

（3）设置设备下次启动的配置文件。

2.5.1 基本概念

涉及配置文件管理的基本概念有 3 个：当前配置、配置文件、下次启动的配置文件。

（1）当前配置

设备内存中的配置信息称为设备的当前配置，它是设备当前正在运行的配置。显然，设备下电后或设备重启时，内存中原有的所有信息（包括配置信息）都会消失。

（2）配置文件

包含设备配置信息的文件称为配置文件，它存在于设备的外部存储器中（注意，不是在内存中），其文件名的格式一般为“*.cfg”或“*.zip”。用户可以将当前配置保存到配置文件中。当设备重启时，配置文件的内容可以被重新加载到内存，成为新的当前配置。配置文件除了具有保存配置信息的作用外，还可以方便设备安装和维护人员查看、备份以及移植配置信息用于其他设备。缺省情况下，保存当前配置时，设备会将配置信息保存到名为“vrpcfg.zip”的配置文件中，并存放于设备的外部存储器的根目录下。

（3）下次启动的配置文件

顾名思义，下次启动的配置文件即为设备下次启动时加载至内存的配置文件。设备重启时，会从指定的配置文件中提取配置信息，并加载至内存中；缺省情况下，下次启动的配置文件的文件名为“vrpcfg.zip”。

2.5.2 保存当前配置

保存当前配置的方式有两种：手动保存和自动保存。

（1）手动保存配置

用户可以使用 **save**[*configuration-file*]命令随时将当前配置以手动方式保存到配置文件中，参数 *configuration-file* 为指定的配置文件名，格式必须为“*.cfg”或“*.zip”。如果未指定配置文件名，则配置文件名缺省为“vrpcfg.zip”。

例如，需要将当前配置保存到文件名为“vrpcfg.zip”的配置文件中时，可进行如下操作。

```
[Huawei] save
 The current configuration will be written to the device.
 Are you sure to continue ? [Y/N] : y              //输入“y”确认保存
```

```
It will take several minutes to save configuration file, please wait...
Configuration file had been saved successfully
Note : The configuration file will take effect after being activated
```

如果还需要将当前配置保存到文件名为“backup.zip”的配置文件中，作为对vrpcfg.zip的备份，则可进行如下操作。

```
[Huawei] save backup.zip
Are you sure to save the configuration to flash : /backup.zip ? [Y/N] : y
Now saving the current configuration to the slot 17.
Save the configuration successfully
```

（2）自动保存配置

自动保存配置功能可以有效降低用户因忘记保存配置而导致配置丢失的风险。自动保存功能分为周期性自动保存和定时自动保存两种方式。

在周期性自动保存方式下，设备会根据用户设定的保存周期，自动完成配置保存；无论设备的当前配置相比配置文件是否有变化，设备都会进行自动保存操作。在定时自动保存方式下，用户设定一个时间点，设备会每天在此时间点自动进行一次保存。缺省情况下，设备的自动保存功能是关闭的，需要用户开启之后才能使用。

周期性自动保存的设置方法如下：首先执行命令 **autosave interval on**，开启设备的周期性自动保存功能，然后执行命令 **autosave interval** *time*，设置自动保存周期。*time* 为指定的时间周期，单位为分钟，默认值为1 440分钟（24小时）。

定时自动保存的设置方法如下：首先执行命令 **autosave time on**，开启设备的定时自动保存功能，然后执行命令 **autosave time** *time-value* ，设置自动保存的时间点。*time-value* 为指定的时间点，格式为hh:mm:ss，默认值为00:00:00。

说明

周期性自动保存与定时自动保存是互斥的。同一时间、同一台设备只允许设置其中一种自动保存方式。如果希望更换自动保存方式，则需要首先取消已经设置的自动保存方式。另外，即使设置了自动保存功能，用户依然可以使用save命令进行手动方式保存配置。

缺省情况下，设备会保存当前配置到“vrpcfg.zip”文件中。如果用户指定了另外一个配置文件作为设备下次启动的配置文件后，则设备会将当前配置保存到新指定的下次启动的配置文件中。

2.5.3 设置下次启动的配置文件

设备支持设置任何一个存在于设备的外部存储器的根目录下（如：flash: /）的“*.cfg”或“*.zip”文件作为设备的下次启动的配置文件。我们可以通过 **startup saved-configuration** *configuration-file* 命令来设置设备下次启动的配置文件，其中 *configuration-file* 为指定配置文件名。如果设备的外部存储器的根目录下没有该配置文件，则系统会提示设置失败。

例如，如果需要指定已经保存的backup.zip文件作为下次启动的配置文件，可执行如下操作。

```
[Huawei] startup saved-configuration backup.zip
This operation will take several minutes, please wait...
Info : Succeeded in setting the file for booting system
```

 **注意**

设置了下次启动的配置文件后，再保存当前配置时，默认会将当前配置保存到所设置的下次启动的配置文件中，从而覆盖了下次启动的配置文件的原有内容。所以，保存当前配置时应该特别小心。

设置好下次启动的配置文件后，一般都会重启设备让配置生效。如果设备是由多人维护的，则很可能出现当前配置信息与下次启动的配置文件中的信息不一致的情况。VRP 系统提供了 **compare configuration** 命令，用来比较当前配置与下次启动的配置文件的差异。执行该命令后，系统会从下次启动的配置文件的首行开始与当前配置进行比较，在比较出不同之处时，将从两者有差异的地方开始显示字符，默认显示 120 个字符。

例如，如果需要比较一下设备的当前配置与之前指定的下次启动的配置文件 backup.zip 之间的差异，则可执行以下操作。

```
[Huawei] compare configuration
The current configuration is not the same as the next startup configuration file.
====== Current configuration line 14 ======
undo http server enable
#
drop illegal-mac alarm
#
vlan batch 10 to 11
#
dot1x enable
mac-authen
#
set transceiver-monitoring disable
====== Configuration file line 14 ======
http server enable
#
drop illegal-mac alarm
#
vlan batch 10 to 11
#
dot1x enable
mac-authen
#
set transceiver-monitoring disable
```

从显示信息中可以看到，当前配置中是取消了 HTTP 服务器功能的（undo http server enable），这一点与下次启动的配置文件是有差异的。

2.6 通过 Telnet 登录设备

学习完本节内容之后，我们应该能够：

（1）了解 Telnet 的基本概念；

（2）通过 Telnet 登录设备。

2.6.1 Telnet 简介

Telnet 协议是 TCP/IP 协议族中应用层协议的一员。Telnet 的工作方式为“服务器/客户端”方式，它提供了从一台设备（Telnet 客户端）远程登录到另一台设备（Telnet 服务器）的方法。Telnet 服务器与 Telnet 客户端之间需要建立 TCP 连接，Telnet 服务器的缺省端口号为 23。

VRP 系统既支持 Telnet 服务器功能，也支持 Telnet 客户端功能。利用 VRP 系统，用户还可以先登录到某台设备，然后将这台设备作为 Telnet 客户端再通过 Telnet 方式远程登录到网络上的其他设备，从而可以更为灵活地实现对网络的维护操作。如图 2-22 所示，路由器 R1 既是 PC 的 Telnet 服务器，又是路由器 R2 的 Telnet 客户端。

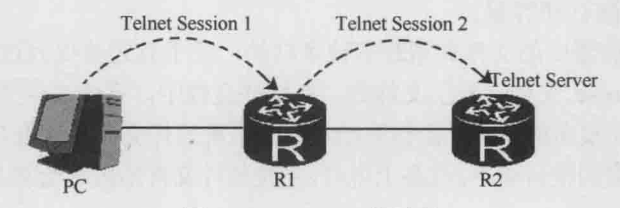


图 2-22 Telnet 二级连接

2.6.2 Telnet 登录设备

邮
电

我们可以在计算机的 Windows 操作系统自带的命令窗口中执行命令 **telnet** *ip-address* 来登录设备。如图 2-23 所示，假设设备的 IP 地址为 10.137.217.177，则输入命令 **telnet** 10.137.217.177 即可。

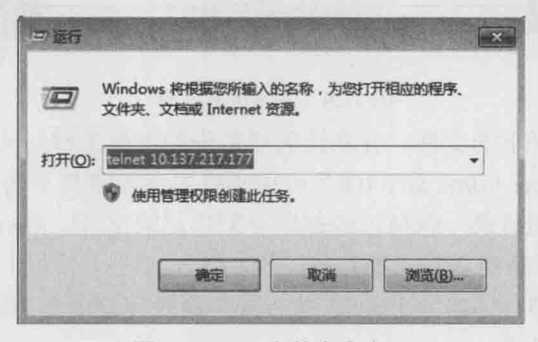


图 2-23 PC 上的命令窗口

单击图 2-23 中的“确定”后，在登录窗口输入登录用户名和密码，验证通过后，出现用户视图的命令行提示符<Huawei>，说明登录设备成功。

2.7 文件管理

VRP 通过文件系统来对设备上的所有文件(包括设备的配置文件、系统软件文件、License 文件、补丁文件等）和目录进行有效的管理。学习完本节内容之后，我们应该能够：

（1）了解文件管理的基本概念；

（2）完成设备的配置文件的备份；

（3）通过 TFTP 和 FTP 实现文件的传输；

（4）删除设备中的文件；

（5）配置设备的启动文件。

2.7.1 基本概念

VRP 文件系统主要用来创建、删除、修改、复制和显示文件及目录，这些文件和目录都存在于设备的外部存储器中。华为路由器支持的外部存储器一般有 Flash 和 SD 卡，交换机支持的外部存储器一般有 Flash 和 CF 卡。除此之外，有的设备还支持通过外接 U 盘来扩充设备的外部存储容量。

设备的外部存储器中的文件类型是多种多样的，除了有之前提到过的配置文件，还有系统软件文件、License 文件、补丁文件等。在这些文件中，系统软件文件具有特别的重要性，因为它其实就是设备的 VRP 操作系统本身。系统软件文件的扩展名为“.cc”，并且必须存放于外部存储器的根目录下。设备上电时，系统软件文件的内容会被加载至内存并运行。

2.7.2 备份配置文件

由于系统升级等原因，我们可能需要将某个设备上的某个配置文件备份到该设备的外部存储器的某个指定文件夹中。下面，我们通过一个例子来说明这一过程。如图 2-24 所示，假设我们已经通过 PC 成功登录到路由器 R1，接下来的步骤将说明如何完成配置文件的备份过程。



图 2-24 备份配置文件

（1）查看当前路径下的文件，并确认需要备份的文件名称与大小

dir [/all] [*filename* | *directory*]命令可用来查看当前路径下的文件，**all** 表示查看当前路径下的所有文件和目录，包括已经删除至回收站的文件。*filename* 表示待查看文件的名称，*directory* 表示待查看目录的路径。

路由器的默认外部存储器为 Flash，执行如下命令可查看路由器 R1 的 Flash 存储器的根目录下的文件和目录。

```
[Huawei] dir
Directory of flash: /

  Idx   Attr    Size (Byte)     Date          Time (LMT)    FileName
    0   -rw-     94,777,088     Jan 19 2013   16:20:29      software.cc
    1   -rw-              0     Jan 28 2013   09:16:34      brdxpon_snmp_cfg.efs
    2   -rw-            396     Jan 28 2013   09:18:27      rsa_host_key.efs
    3   -rw-          1,317     Mar 20 2013   10:22:32      private-data.txt
    4   -rw-         44,192     Mar 20 2013   10:26:25      mon_file.txt
    5   -rw-            540     Jan 28 2013   09:18:26      rsa_server_key.efs
    6   drw-              -     Jun 21 2012   10:25:25      cdr
    7   -rw-          1,351     Mar 08 2013   13:55:28      vrpcfg.zip
```

```
    8  -rw-        7,301,397      Jan 28 2013      09:18:26        abcd.zip
    9  drw-                -      Aug 21 2012      11:21:58        dhcp

217,168 KB total （94,104 KB free）
<Huawei>
```

从回显信息中，我们看到了名为“vrpcfg.zip”的配置文件，大小为1 351字节，假设它就是我们需要备份的配置文件。

（2）新建目录

创建目录的命令为 **mkdir** *directory*，*directory* 表示需要创建的目录。在Flash的根目录下创建一个名为backup的目录。

```
[Huawei] mkdir flash : /backup
Info : Create directory flash : /backup......Done
```

（3）复制并重命名文件

复制文件的命令为 **copy** *source-filename destination-filename*，*source-filename* 表示被复制文件的路径及源文件名，*destination-filename* 表示目标文件的路径及目标文件名。

把需要备份的配置文件 vrpcfg.zip 复制到新目录 backup 下，并重命名为vrpcfgbak.zip。

```
[Huawei] copy vrpcfg.zip flash : /backup/vrpcfgbak.zip
Copy flash : /vrpcfg.zip to flash : /backup/vrpcfgbak.zip ? （y/n）[n] : y
100%   complete
Info : Copied file flash : /vrpcfg.zip to flash : /backup/vrpcfgbak.zip...Done
```

（4）查看备份后的文件

cd *directory* 命令用来修改当前的工作路径。我们可以执行如下操作来查看文件备份是否成功。

```
[Huawei] cd flash : /backup
[Huawei] dir
Directory of flash : /backup/

  Idx  Attr    Size（Byte）  Date           Time（LMT）   FileName
    0  -rw-          1,351   Mar 20 2013    14:36:15      vrpcfgbak.zip

217,168 KB total （94,072 KB free）
<Huawei>
```

回显信息表明，backup目录下已经有了文件vrpcfgbak.zip，配置文件vrpcfg.zip的备份过程已顺利完成。

2.7.3　传输文件

1. 通过TFTP传输文件

TFTP（Trivial File Transfer Protocol，简单文件传输协议）是TCP/IP协议族中应用层协议的一员，它是一种简单的文件传输协议，其传输层协议是UDP，端口号为69。使用TFTP来传输文件时，无需进行用户名和密码的验证，也不会对数据进行加密。当需要传输的文件较小，同时对网络安全环境放心的情况下，我们可以选择通过TFTP来传输文件，这样可以简化操作。

TFTP的工作方式为“服务器/客户端”方式，华为的交换机和路由器仅支持作为TFTP客户端。如图2-25所示，PC作为TFTP服务器，路由器作为TFTP客户端，我们的任务

是要将 PC 上的某个 VRP 系统软件文件传输到路由器上。

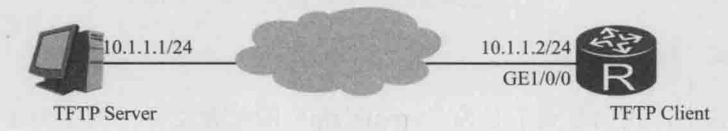


图 2-25 通过 TFTP 进行文件传输

利用 TFTP 进行文件传输的命令是 **tftp** *tftp-server*{**get**|**put**}*source-filename* [*destination-filename*]，其中 *tftp-server* 表示 TFTP 服务器的 IP 地址，**get** 表示从 TFTP 服务器下载文件到 TFTP 客户端，**put** 表示从 TFTP 客户端上传文件到 TFTP 服务器，*source-filename* 表示源文件名，*destination-filename* 表示目标文件名。现在，我们需要将 PC 上的 VRP 系统软件文件 devicesoft.cc 下载到路由器上，执行的操作如下。

```
[Huawei] tftp 10.1.1.1 get devicesoft.cc
Info : Transfer file in binary mode.
Downloading the file from the remote TFTP server. Please wait...\
TFTP : Downloading the file successfully.
93832832 bytes received in 722 seconds.
```

使用 TFTP 虽然简单方便，但是安全性较差，任何用户都可以接入 TFTP 服务器进行上传和下载文件。为提高安全性，我们还可以使用 FTP 来进行文件传输。

2. 通过 FTP 传输文件

FTP（File Transfer Protocol，文件传输协议）也是 TCP/IP 协议族中应用层协议的一员，其传输层协议是 TCP，端口号为 21，工作方式也是“服务器/客户端”方式。基于 VRP 系统的交换机和路由器既可以作为 FTP 客户端，又可以作为 FTP 服务器。建立 FTP 连接需要进行用户名和密码的验证，安全性得到了一定的保障，同时 FTP 协议还支持对服务器进行文件删除、文件目录创建和删除等操作。

如图 2-26 所示，PC 作为 FTP 服务器，路由器作为 FTP 客户端，我们的任务是要将 PC 上的某个 VRP 系统软件文件传输到路由器上。



图 2-26 通过 FTP 进行文件传输

命令 **ftp** *host-ip* [*port-number*]是用来建立 FTP 连接的，其中，*host-ip* 表示 FTP 服务器的 IP 地址；*port-number* 表示 FTP 服务器的端口号，默认为 21。

```
[Huawei] ftp 10.1.1.1
Trying 10.1.1.1 ...
Press CTRL+K to abort
Connected to 10.1.1.1.
220 FTP service ready.
User（10.1.1.1:（none））: admin        //服务器的用户名
331 Password required for admin.
Enter password:                       //服务器的密码
230 User logged in.

[Huawei-ftp]
```

接下来，执行命令 **dir**，查看 FTP 服务器上都有哪些文件，以便选取需要传输的系统软件文件。

```
[Huawei-ftp] dir
200 Port command successful.
150 Opening data connection for directory list.
drw-rw-rw-    1 ftp      ftp             0    Apr 17 10：53 back
drw-rw-rw-    1 ftp      ftp             0    Apr 17 10：53 backup
-rwxrwxrwx    1 noone    nogroup         0    Mar 23 15：49 aaa.cfg
-rwxrwxrwx    1 noone    nogroup      1351    Apr 02 20：37 vrpcfgbak.zip
-rwxrwxrwx    1 noone    nogroup    286620    Apr 07 08：56 sacrule.dat
-rw-rw-rw-    1 ftp      ftp      93832832    Mar 30 18：29 vrpsoft.cc
8 File sent ok
FTP : 734 byte（s） received in 0.129 second（s） 5.68Kbyte（s）/sec.

[Huawei-ftp]
```

在 FTP 中，**get** 和 **put** 是对文件的两种不同操作方式。与 TFTP 一样，**get** 表示从 FTP 服务器下载文件到 FTP 客户端，命令格式为 **get** *source-filename* [*destination-filename*]，而 **put** 表示从 FTP 客户端上传文件到 FTP 服务器，命令格式为 **put** *source-filename* [*destination-filename*]。

本例中，命令 **get** *vrpsoft.cc devicesoft.cc* 表示从 FTP 服务器（PC）下载名为 vrpsoft.cc 的系统软件到路由器上，并重新以 devicesoft.cc 为文件名进行保存。

```
[Huawei-ftp] get vrpsoft.cc devicesoft.cc
200 Port command okay.
150 Opening ASCII mode data connection for vrpsoft.cc.
226 Transfer complete.
FTP : 93832832 byte（s） received in 722 second（s）560.70byte（s）/sec.
```

FTP 协议仍然是采用明文进行数据传输。为进一步提高安全性，我们还可以通过 SFTP（Secure File Transfer Protocol，安全文件传输协议）来传输文件。SFTP 可以对传输数据进行严格的加密和完整性保护，安全性非常高。

2.7.4 删除文件

当设备的外部存储器的可用空间不够时，我们就很可能需要删除其中的一些无用文件。删除文件的命令为 **delete** [**/unreserved**] [**/force**] *filename*，其中**/unreserved** 表示彻底删除指定文件，删除的文件将不可恢复；**/force** 表示无需确认直接删除文件；*filename* 表示要删除的文件名。

如果不使用**/unreserved**，则 **delete** 命令删除的文件将被保存到回收站中，而使用 **undelete** 命令则可恢复回收站中的文件。注意，保存到回收站中的文件仍然会占用存储器空间。**reset recycle-bin** 命令将会彻底删除回收站中的所有文件，这些文件将被永久删除，不能再被恢复。

例如，如果我们已经确定设备上的文件 abcd.zip 不再有用，需要彻底删除，则可进行如下操作。

```
[Huawei] delete /unreserved abcd.zip
Warning : The contents of file flash：/backup/abcd.zip cannot be recycled. Continue ?
（y/n）[n] : y
Info : Deleting file flash : /backup/abcd.zip...
```

```
Deleting file permanently from flash will take a long time if needed......succeed.
```

2.7.5 设置系统启动文件

所谓启动文件，是指设备在启动时，需要从系统外部存储器中加载至内存并运行的系统软件文件及其他相关文件。在设置下次启动使用的启动文件之前，可以先执行 **display startup** 命令查看设备当前设置的下次启动时所使用的启动文件情况。

```
[Huawei] display startup
MainBoard:
  Startup system software:                    flash:/software.cc
  Next startup system software:               flash:/software.cc
  Backup system software for next startup:    null
  Startup saved-configuration file:           flash:/vrpcfg.zip
  Next startup saved-configuration file:      flash:/vrpcfg.zip
  Startup license file:                       null
  Next startup license file:                  null
  Startup patch package:                      null
  Next startup patch package:                 null
  Startup voice-files:                        null
  Next startup voice-files:                   null
```

显示信息表明，设备下次启动时将使用的系统软件文件是 software.cc。设置下次启动使用的系统软件文件的命令为 **startup system-software** *system-file*，*system-file* 表示指定的系统软件文件名。例如，如果需要将 devicesoft.cc 设置为下次启动时使用的系统软件文件，可执行如下操作。

```
[Huawei] startup system-software devicesoft.cc
This operation will take several minutes, please wait...
Info: Succeeded in setting the file for booting system
```

然后再次执行 **display startup** 命令，检查设置是否成功。

```
[Huawei] display startup
MainBoard:
  Startup system software:                    flash:/software.cc
  Next startup system software:               flash:/devicesoft.cc
  Backup system software for next startup:    null
  Startup saved-configuration file:           flash:/vrpcfg.zip
  Next startup saved-configuration file:      flash:/vrpcfg.zip
  Startup license file:                       null
  Next startup license file:                  null
  Startup patch package:                      null
  Next startup patch package:                 null
  Startup voice-files:                        null
  Next startup voice-files:                   null
```

从回显信息中我们可以看到，下次启动时将使用的系统软件文件已经成功设置成了 devicesoft.cc。

2.8 基础配置常用命令

VRP 命令的总数达数千条之多，其中一些命令的使用频率非常高，并且涉及系统的基础配置。表 2-4 列出了一些与 VRP 基础配置相关的常用命令，读者应首先学会并熟悉

这些命令，然后再逐步学习和了解其他命令的使用方法。

表 2-4 **VRP 基础配置常用命令**

命令格式	简要说明
authentication-mode { **aaa** | **password** | **none** }	设置登录用户界面的验证方式
autosave interval { *value* | *time* | ***configuration*** *time* }	设置周期性自动保存当前配置
autosave time { *value* | *time-value* }	设置定时自动保存当前配置
cd *directory*	修改用户当前的工作路径
clock datetime *HH: MM: SS YYYY-MM-DD*	设置当前日期和时钟
clock timezone *time-zone-name* { **add** | **minus** } *offset*	设置本地时区信息
compare configuration [*configuration-file*] [*current-line-number save-line-number*]	比较当前配置与下次启动的配置文件内容
copy *source-filename destination-filename*	复制文件
delete [**/unreserved**] [**/force**] { *filename* | *devicename* }	删除文件
dir [**/all**] [*filename* | *directory*]	显示文件和目录
display current-configuration	查看当前生效的配置信息
display this	查看当前视图的运行配置
display startup	查看启动文件信息
display user-interface [*ui-type ui-number1* | *ui-number*] [**summary**]	查看用户界面信息
ftp *host-ip* [*port-number*]	与 FTP 服务器建立连接
get *source-filename* [*destination-filename*]	从服务器下载文件到客户端
local-user *user-name* **password cipher** *password*	创建本地用户，并设置密码
local-user *user-name* **service-type telnet**	配置本地用户的接入类型
mkdir *directory*	创建新的目录
move *source-filename destination-filename*	将源文件从指定目录移动到目标目录中
put *source-filename* [*destination-filename*]	从客户端上传文件到服务器
quit	从当前视图退回到上一层视图。如果当前视图为用户视图，则退出系统
reboot	重新启动设备
reset recycle-bin	彻底删除当前目录下回收站中的内容
save	保存当前配置信息
schedule reboot { **at** *time* | **delay** *interval* }	配置设备的定时重启功能
startup saved-configuration *configuration-file*	设置系统下次启动时使用的配置文件
sysname *host-name*	设置设备的主机名
system-view	该命令用来使用户从用户视图进入系统视图
telnet *host-name* [*port-number*]	从当前设备使用 Telnet 协议登录到其他设备
tftp *tftp-server* { **get** | **put** } *source-filename* [*destination-filename*]	上传文件到 TFTP 服务器，或从 TFTP 服务器下载文件
user-interface [*ui-type*] *first-ui-number* [*last-ui-number*]	进入一个用户界面视图或多个用户界面视图
user-interface maximum-vty *number*	设置登录用户的最大数目
user privilege level *level*	设置用户级别

2.9 练习题

1．（多选）VRP 是一种（ ）。

A．网络操作系统 B．系统软件

C．网络设备　　D．支撑多种网络设备的软件平台

2．（单选）以下哪个提示符表示的是接口视图？（　）

A．<Huawei>　　B．[Huawei]

C．[Huawei-GigabitEthernet0/0/1]　　D．[Huawei-Vlan1]

3．（单选）级别是 2 级的用户可以操作什么级别的 VRP 命令？（　）

A．0 级和 1 级　　B．0 级、1 级和 2 级

C．2 级　　D．0 级、1 级、2 级和 3 级

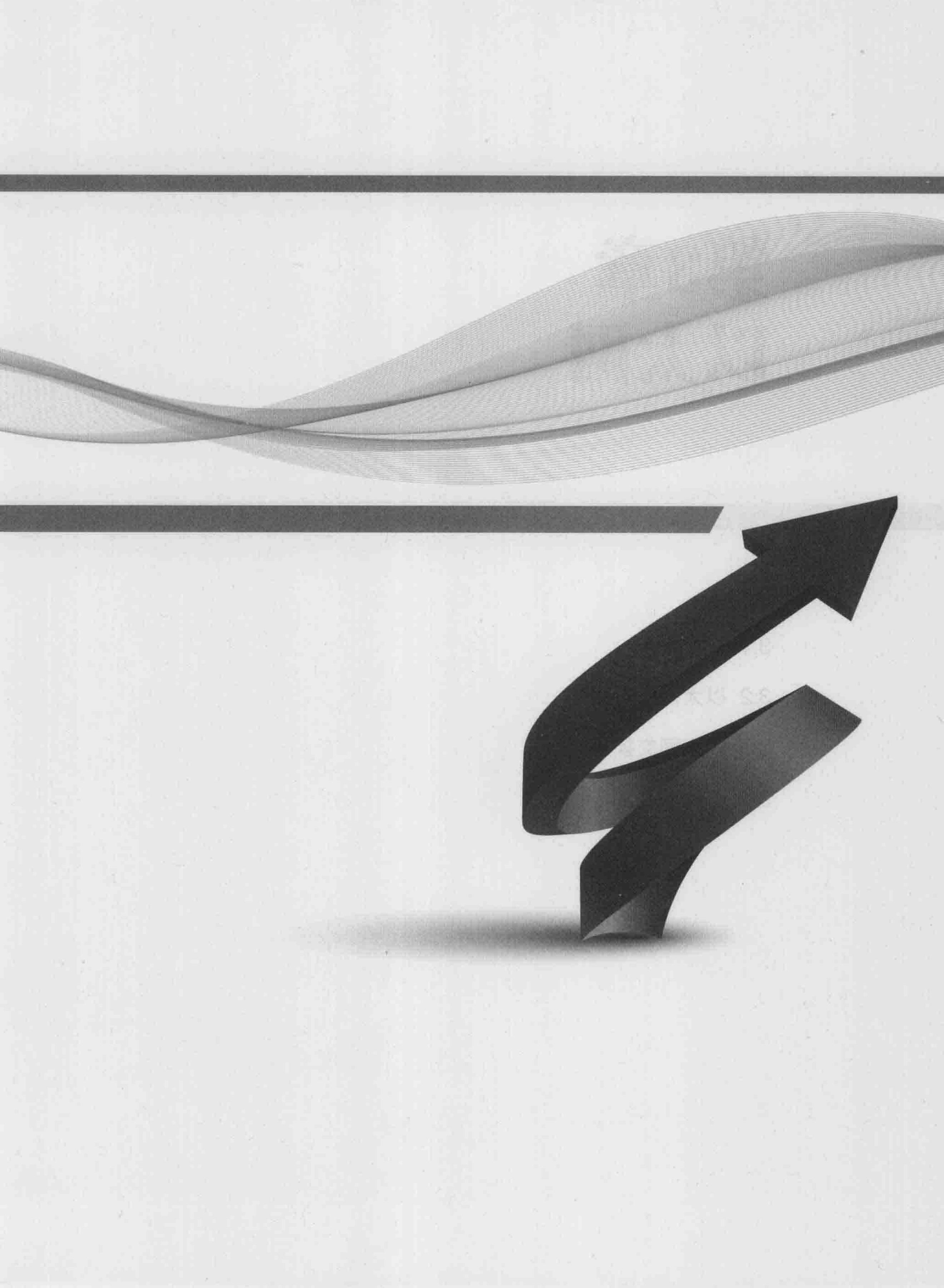
4．（单选）FTP 的缺省端口号是？（　）

A．23　　B．69　　C．21　　D．24

5．（多选）首次登录新设备后，一般需要进行哪些配置？（　）

A．配置设备名称　　B．配置系统时钟

C．配置设备的 IP 地址　　D．配置用户界面



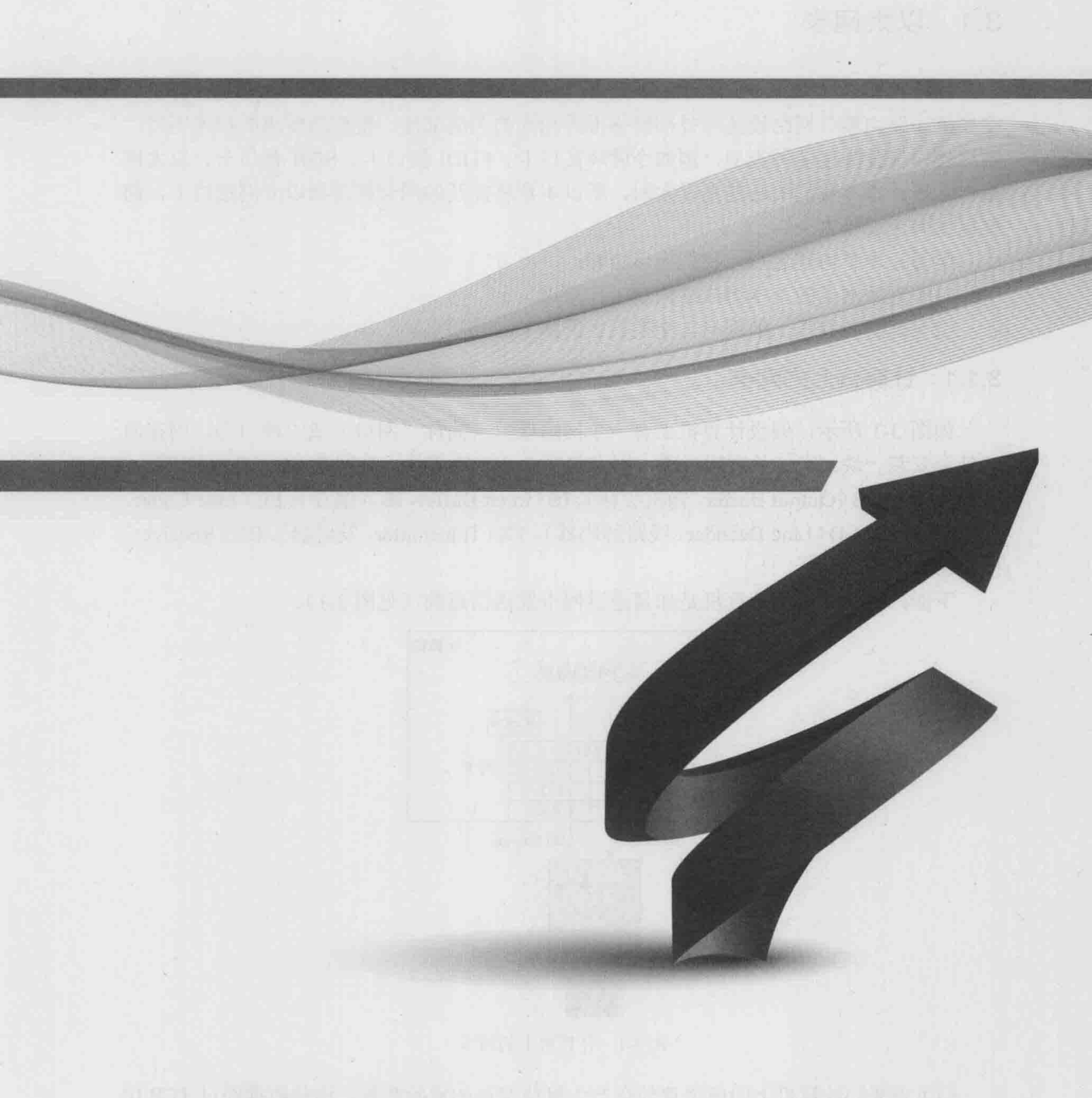
第3章 以太网

3.1 以太网卡

3.2 以太网帧

3.3 以太网交换机

3.4 ARP



3.1 以太网卡

网络接口卡（Network Interface Card，NIC）通常也简称为“网卡”，它是计算机、交换机、路由器等网络设备与外部网络世界相连的关键部件。根据所使用的技术不同，网络接口卡分为很多种类型，例如令牌环接口卡、FDDI 接口卡、SDH 接口卡、以太网接口卡等。本章我们关心的是以太网，所以本章所提及的网卡都是指以太网接口卡，简称以太网卡或以太卡。

学习完本节内容之后，我们应该能够：

（1）理解网卡的基本组成结构和工作原理；

（2）理解计算机上的网卡与交换机上的网卡的异同点。

3.1.1 计算机上的网卡

如图 3-1 所示，假设计算机上有一个网络接口（简称“网口”或“端口”），则在网口处会安装一块网卡。从逻辑上讲，网卡包含 7 个功能模块，分别是 CU（Control Unit，控制单元）、OB（Output Buffer，输出缓存）、IB（Input Buffer，输入缓存）、LC（Line Coder，线路编码器）、LD（Line Decoder，线路解码器）、TX（Transmitter，发射器）、RX（Receiver，接收器）。

下面，我们来看看计算机是如何通过网卡发送信息的（见图 3-1）。

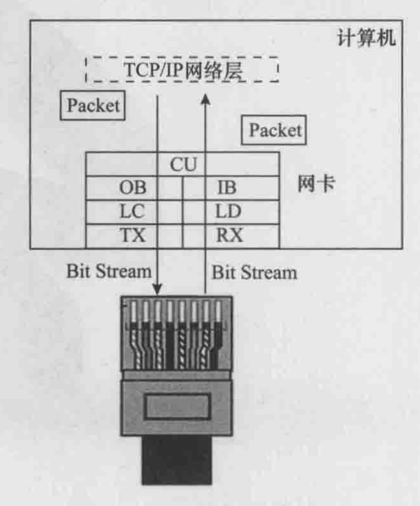


图 3-1 计算机上的网卡

（1）首先，计算机上的应用软件会产生等待发送的原始数据，这些数据经过 TCP/IP 模型的应用层、传输层、网络层处理后，得到一个一个的数据包 （Packet）。然后，网络层会将这些数据包逐个下传给网卡的 CU。

（2）CU 从网络层那里接收到数据包之后，会将每个数据包封装成帧（Frame）。因为本章所说的网卡都是指以太网卡，所以封装成的帧都是以太帧（Ethernet Frame）。然后，CU 会将这些帧逐个传递给 OB。

（3）OB 从 CU 那里接收到帧后，会按帧的接收顺序将这些帧排成一个队列，然后

将队列中的帧逐个传递给 LC。先从 CU 那里接收到的帧会被先传递给 LC。

（4）LC 从 OB 那里接收到帧后，会对这些帧进行线路编码。从逻辑上讲，一个帧就是长度有限的一串“0”和“1”。OB 中的“0”和“1”所对应的物理量（指电平、电流、电荷等）只适合于待在缓存中，而不适合于在线路（传输介质，例如双绞线）上进行传输。LC 的作用就是将这些“0”和“1”所对应的物理量转换成适合于在线路上进行传输的物理信号（指电流/电压波形等），并将物理信号传递给 TX。

（5）TX 从 LC 那里接收到物理信号后，会对物理信号的功率等特性进行调整，然后将调整后的物理信号通过线路（例如双绞线）发送出去。

再来看看计算机是如何通过网卡接收信息的（见图 3-1）。

（1）首先，RX 从传输介质（例如双绞线）那里接收到物理信号（指电流/电压波形等），然后对物理信号的功率等特性进行调整，再将调整后的物理信号传递给 LD。

（2）LD 会对来自 RX 的物理信号进行线路解码。所谓线路解码，就是从物理信号中识别出逻辑上的“0”和“1”，并将这些“0”和“1”重新表达为适合于待在缓存中的物理量（指电平、电流、电荷等），然后将这些“0”和“1”以帧为单位逐个传递给 IB。

（3）IB 从 LD 那里接收到帧后，会按帧的接收顺序将这些帧排成一个队列，然后将队列中的帧逐个传递给 CU。先从 LD 那里接收到的帧会被先传递给 CU。

（4）CU 从 IB 那里接收到帧后，会对帧进行分析和处理。一个帧的处理结果有且只有两种可能：直接将这个帧丢弃，或者将这个帧的帧头和帧尾去掉，得到数据包，然后将数据包上传给 TCP/IP 模型的网络层。

（5）从 CU 上传到网络层的数据包会经过网络层、传输层、应用层逐层处理，处理后的数据被送达给应用软件使用。当然，数据也可能会在某一层的处理过程中被提前丢弃了，从而无法送达给应用软件。

3.1.2 交换机上的网卡

如图 3-2 所示，一台交换机上总是有多个用来转发数据的网络接口（简称“网口”或“端口”），每个转发数据的网口都有一块网卡与之相对应，不同的网口对应不同的网卡。本章我们关心的是以太网，所以这里所说的交换机是指以太网交换机，也就是说，交换机上每个转发数据的网口所使用的网卡都是以太网卡。

比较图 3-2 和图 3-1 可以发现，交换机上的网卡和计算机上的网卡在组成结构上是完全一样的，都是由 CU、OB、IB、LC、LD、TX、RX 这 7 个功能模块组成。

下面，我们来看看交换机上的网卡是如何转发数据的。

转发数据分为转入数据和转出数据，先来看看网卡是如何转入数据的（请见图 3-2 中中间的那块网卡）。

（1）首先，RX 从传输介质（例如双绞线）那里接收到物理信号（指电流/电压波形等），然后对物理信号的功率等特性进行调整，再将调整后的物理信号传递给 LD。这个过程与计算机上网卡的 RX 的工作过程完全一样。

（2）LD 的工作过程与计算机上网卡的 LD 的工作过程完全一样，这里不再赘述。

（3）IB 的工作过程与计算机上网卡的 IB 的工作过程完全一样，这里不再赘述。

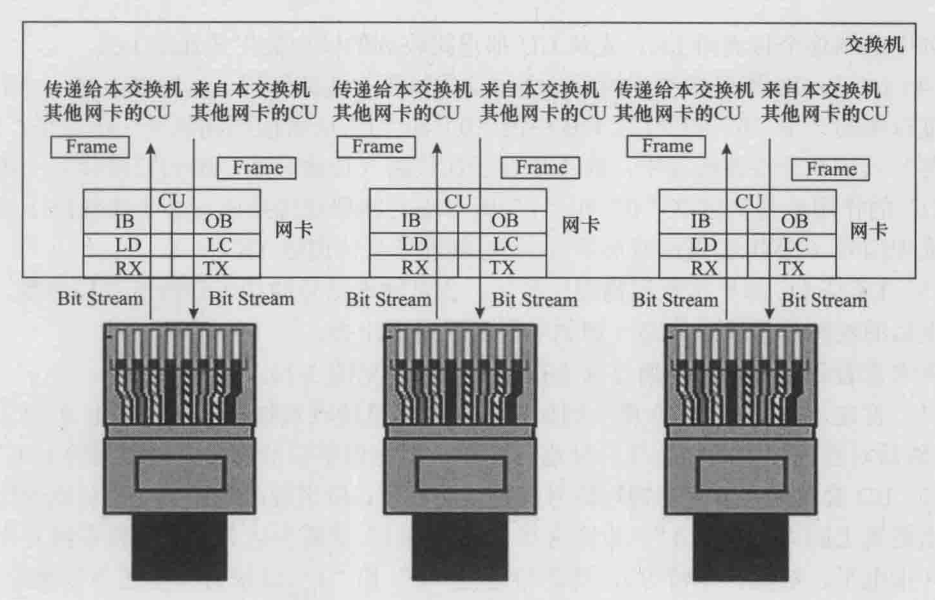


图 3-2 交换机上的网卡

（4）CU 从 IB 那里接收到帧后，会对帧进行分析和处理。一个帧的处理结果有且只有 3 种可能：或者被直接丢弃；或者被传递给本交换机的其他某一块网卡的 CU；或者被复制成 n 个帧，然后将这 n 个帧分别传递给本交换机的其他 n 个网卡的 CU，每个 CU 得到一个帧。

我们再来看看网卡是如何转出数据的（请见图 3-2 中中间的那块网卡）。

（1）与计算机上网卡的 CU 不同，交换机上网卡的 CU 是直接从本交换机的其他网卡的 CU 那里接收到帧的，然后 CU 会将这些帧传递给 OB。

（2）OB 的工作过程与计算机上网卡的 OB 的工作过程完全一样，这里不再赘述。

（3）LC 的工作过程与计算机上网卡的 LC 的工作过程完全一样，这里不再赘述。

（4）TX 的工作过程与计算机上网卡的 TX 的工作过程完全一样，这里不再赘述。

至此，我们描述了计算机上的网卡是如何收发数据的，以及交换机上的网卡是如何转发数据的。从这些描述中，还可以总结出以下几个知识点。

（1）网卡工作在 TCP/IP 模型的数据链路层和物理层，同时具有数据链路层的功能和物理层的功能。

（2）计算机上的网卡是用来收发数据的，交换机上的网卡是用来转发数据的。

（3）交换机上的网卡和计算机上的网卡在组成结构上是完全一样的，都是由 CU、OB、IB、LC、LD、TX、RX7 个功能模块组成的。

（4）除了 CU 外，交换机上网卡和计算机上网卡的各个功能模块的工作过程完全一样。

（5）计算机上网卡的 CU 需要进行帧（Frame）的封装和解封装，并与计算机上 TCP/IP 模型的网络层交换数据包（Packet）。交换机上网卡的 CU 不需要进行帧的封装和解封装，而是直接与本交换机上其他网卡的 CU 进行帧的交换。

不管是在计算机上也好，还是在交换机上也好，一个端口总是对应一块网卡（或者

说一个端口总是拥有一块属于自己的网卡），不同的端口对应不同的网卡。网卡的作用就是用来进行数据的收发或转发。当我们说某个端口在收发或转发数据时，实质上是指这个端口的网卡在收发或转发数据。

最后需要说明的是，通常情况下，如果一台计算机上有多个端口（网口），则这些端口的网卡都是以独立器件的形式出现的，并且每块网卡被安装在自己所对应的那个端口的位置。而在交换机上，网卡通常是以集成芯片的形式出现的。比如，一台拥有8个端口的交换机内可能只有2块集成芯片，其中一块集成芯片上集成了4块网卡，这4块网卡分别对应交换机的4个端口；而另一块集成芯片上也集成了4块网卡，这4块网卡分别对应交换机的另外4个端口。这2块集成芯片在交换机内的空间位置并不重要。

3.1.3 练习题

1. （多选）下列描述中正确的是？（ ）
   A. 网卡工作在数据链路层，只具有数据链路层的功能
   B. 网卡工作在数据链路层和物理层，同时具有数据链路层的功能和物理层的功能
   C. 网卡的线路编/解码器只具有数据链路层的功能
   D. 网卡的线路编/解码器只具有物理层的功能
2. （多选）下列描述中错误的是？（ ）
   A. 网卡的TX/RX只具有数据链路层的功能
   B. 网卡的TX/RX只具有物理层的功能
   C. 网卡是不可能与TCP/IP模型的网络层交换数据的
   D. 网卡有可能会与TCP/IP模型的网络层交换数据
3. （多选）下列描述中正确的是？（ ）
   A. 当我们说某个端口是以太口时，其实是指这个端口的网卡是一块以太网卡
   B. 计算机上的网卡需要进行帧的封装和解封装
   C. 计算机上不可能有多个网口，因为计算机不是用来转发数据的设备
   D. 交换机上总是有多个网口，因为交换机就是用来转发数据的设备
4. （多选）下列描述中错误的是？（ ）
   A. 一块网卡只能控制一个网口的数据收发/转发行为
   B. 一块网卡可以同时控制多个网口的数据收发/转发行为
   C. 多块网卡可以同时控制一个网口的数据收发/转发行为

3.2 以太网帧

以太网技术使用的帧是以太网帧，令牌环技术使用的帧是令牌环帧，FR技术使用的帧是FR帧，如此等等。本书中所提到的帧，如无特别说明，都是指以太网帧。

学习完本节内容之后，我们应该能够：

（1）理解、熟悉并记住MAC地址的结构和分类；

（2）理解、熟悉并记住 Ethernet II 格式的以太帧的结构；

（3）分清楚单播 MAC 地址、组播 MAC 地址、广播 MAC 地址、单播帧、组播帧、广播帧这几个概念的区别与联系。

3.2.1 MAC 地址

1980 年 2 月，美国电气和电子工程师协会（IEEE）召开了一次会议，此次会议启动了一个庞大的技术标准化项目，称为 IEEE 802 项目（IEEE Project 802）。802 中的“80”是指 1980 年，“2”是指 2 月份。

IEEE 802 项目旨在制定一系列的关于局域网（LAN）的标准。以太网标准（IEEE 802.3）、令牌环网络标准（IEEE 802.5）、令牌总线网络标准（IEEE 802.4）等局域网标准都是 IEEE 802 项目的成果。我们把 IEEE 802 项目所制定的各种标准统称为 IEEE 802 标准。

MAC（Medium Access Control）地址是在 IEEE 802 标准中定义并规范的，凡是符合 IEEE 802 标准的网络接口卡（如以太网卡、令牌环网卡等）都必须拥有一个 MAC 地址。注意，不是任何一块网络接口卡都必须拥有 MAC 地址。例如，SDH 网络接口卡就没有 MAC 地址，因为这种接口并不遵从 IEEE 802 标准。顺便强调一下，以下所说的网卡，都是指以太网卡。

如同每个人都有一个身份证号码来标识自己一样，每块网卡也拥有一个用来标识自己的号码，这个号码就是 MAC 地址，其长度为 48bit（6 个字节）。不同的网卡，其 MAC 地址也不相同。也就是说，一块网卡的 MAC 地址是具有全球唯一性的。

一个制造商在生产制造网卡之前，必须先向 IEEE 注册，以获取到一个长度为 24bit（3 个字节）的厂商代码，也称为 OUI（Organizationally-Unique Identifier）。制造商在生产制造网卡的过程中，会往每一块网卡中的 ROM（Read Only Memory）中烧入一个 48bit 的 BIA 地址（Burned-In Address，固化地址），BIA 地址的前 3 个字节就是该制造商的 OUI，后 3 个字节由该制造商自己确定，但不同的网卡，其 BIA 地址的后 3 个字节不能相同。烧入进网卡的 BIA 地址是不能被更改的，只能被读取出来使用。图 3-3 显示了 BIA 地址的格式。

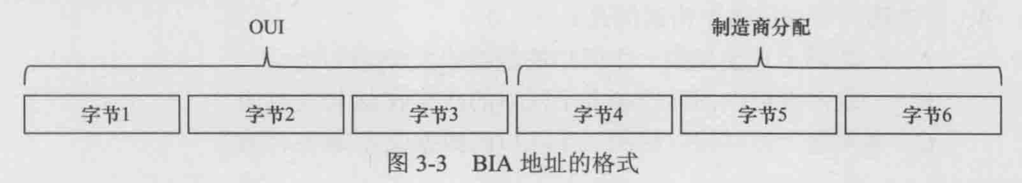


图 3-3　BIA 地址的格式

注意，BIA 地址只是 MAC 地址的一种，更准确地说，BIA 地址是一种单播 MAC 地址。MAC 地址共分为 3 种，分别为单播 MAC 地址、组播 MAC 地址、广播 MAC 地址。这 3 种 MAC 地址的定义分别如下（见图 3-4）。

（1）单播 MAC 地址是指第一个字节的最低位是 0 的 MAC 地址。

（2）组播 MAC 地址是指第一个字节的最低位是 1 的 MAC 地址。

（3）广播 MAC 地址是指每个比特都是 1 的 MAC 地址。广播 MAC 地址是组播 MAC 地址的一个特例。

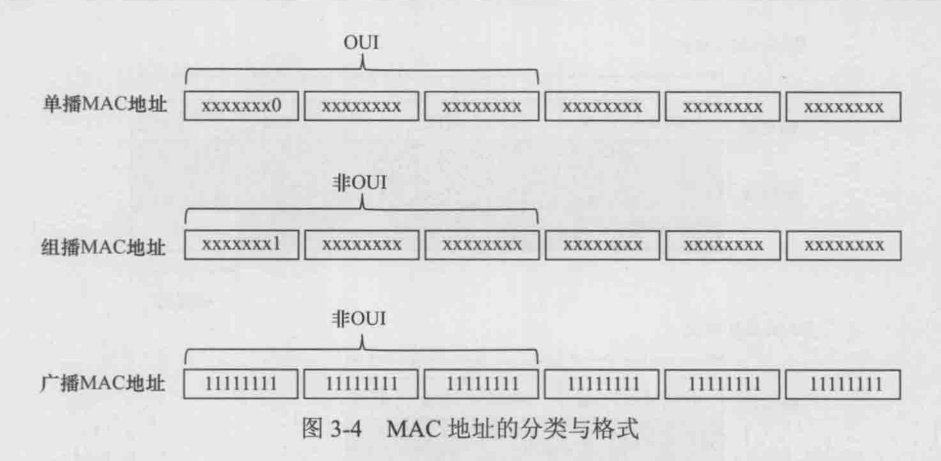


图 3-4 MAC 地址的分类与格式

一个单播 MAC 地址（例如 BIA 地址）标识了一块特定的网卡；一个组播 MAC 地址标识的是一组网卡；广播 MAC 地址是组播 MAC 地址的一个特例，它标识了所有的网卡。

从图 3-4 我们可以发现，并非任何一个 MAC 地址的前 3 个字节都是 OUI，只有单播 MAC 地址的前 3 个字节才是 OUI，而组播或广播 MAC 地址的前 3 个字节一定不是 OUI。特别需要说明的是，OUI 的第一个字节的最低位一定是 0。

一个 MAC 地址有 48bit，为了方便起见，通常采用十六进制数的方式来表示一个 MAC 地址：每两位十六进制数 1 组（即 1 个字节），一共 6 组，中间使用中划线连接；也可以每四位十六进制数 1 组（即 2 个字节），一共 3 组，中间使用中划线连接。图 3-5 对这两种表示方法进行了举例说明。



图 3-5 MAC 地址的表示方法

3.2.2 以太帧的格式

以太网技术所使用的帧称为以太网帧（Ethernet Frame），或简称以太帧。以太帧的格式有两个标准：一个是由 IEEE 802.3 定义的，称为 IEEE 802.3 格式；一个是由 DEC（Digital Equipment Corporation）、Intel、Xerox 这三家公司联合定义的，称为 Ethernet II 格式，也称为 DIX 格式。以太帧的两种格式如图 3-6 所示。虽然 Ethernet II 格式与 IEEE 802.3 格式存在一定的差别，但它们都可以应用于以太网。目前的网络设备都可以兼容这两种格式的帧，但 Ethernet II 格式的帧使用得更加广泛些。通常，承载了某些特殊协议信息的以太帧才使用 IEEE 802.3 格式，而绝大部分的以太帧使用的都是 Ethernet II 格式。

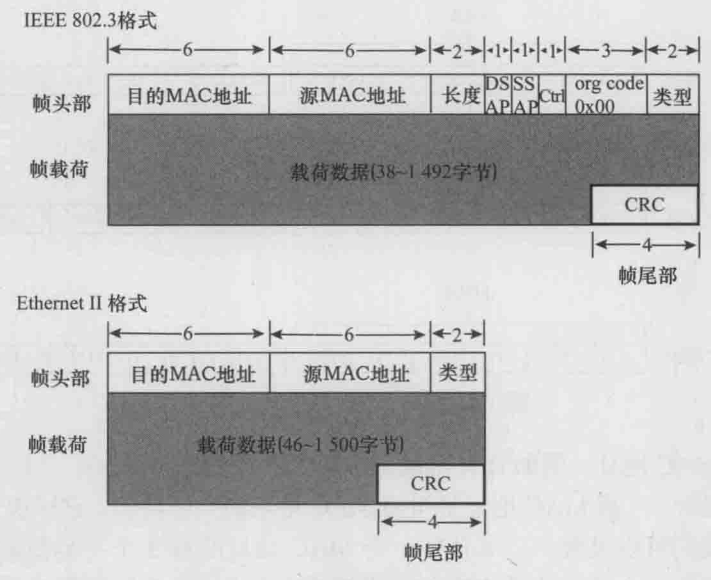


图 3-6　以太帧的两种标准格式

下面是关于 Ethernet II 格式的以太帧中各个字段的描述。

（1）目的 MAC 地址：该字段有 6 个字节，用来表示该帧的接收者（目的地）。目的 MAC 地址可以是一个单播 MAC 地址，或一个组播 MAC 地址，或一个广播 MAC 地址。

（2）源 MAC 地址：该字段有 6 个字节，用来表示该帧的发送者（出发地）。源 MAC 只能是一个单播 MAC 地址。

（3）类型：该字段有 2 个字节，用来表示载荷数据的类型。例如，如果该字段的值是 0x0800，则表示载荷数据是一个 IPv4 Packet；如果该字段的值是 0x86dd，则表示载荷数据是一个 IPv6 Packet；如果该字段的值是 0x0806，则表示载荷数据是一个 ARP Packet；如果该字段的值是 0x8848，则表示载荷数据是一个 MPLS 报文，如此等等。

（4）载荷数据：该字段的长度是可变的，最短为 46 字节，最长为 1 500 字节，它是该帧的有效载荷，载荷的类型由前面的类型字段表示。

（5）CRC 字段：该字段有 4 个字节。CRC 的全称是 Cyclic Redundancy Check，它的作用是对该帧进行检错校验，其具体的工作机制描述已超出了本书的知识范围，所以这里略去不讲。

IEEE 802.3 格式的以太帧中，目的 MAC 地址字段、源 MAC 地址字段、类型字段、载荷数据字段、CRC 字段的功能和作用与 Ethernet II 格式是一样的，这里不再赘述。关于其他几个字段（长度字段、DSAP 字段等）的描述，已经超出了本书的知识范围，所以这里略去不讲。

需要特别说明的是，根据目的 MAC 地址的种类不同，以太帧可以分为以下 3 种不同的类型。

（1）单播以太帧（或简称单播帧）：目的 MAC 地址为一个单播 MAC 地址的帧。

（2）组播以太帧（或简称组播帧）：目的 MAC 地址为一个组播 MAC 地址的帧。

（3）广播以太帧（或简称广播帧）：目的 MAC 地址为广播 MAC 地址的帧。

3.2.3　练习题

1. （单选）一个网卡制造商利用同一个 OUI 最多可以生产多少块网卡？（　）
   A. 1 块　　B. 16 777 216 块　　C. 256 块　　D. 65 536 块
2. （单选）MAC 地址 05-1e-10-0d-d0-03 是哪个？（　）
   A. 单播 MAC 地址
   B. 组播 MAC 地址
   C. 广播 MAC 地址
3. （多选）下列描述中正确的是？（　）
   A. 以太帧的格式有两种，分别是 IEEE 802.3 格式和 IEEE 802.4 格式
   B. 以太帧的格式有两种，分别是 IEEE 802.3 格式和 Ethernet II 格式
   C. 以太帧的格式有两种，分别是 Ethernet I 格式和 Ethernet II 格式
   D. 以太帧的格式有两种，分别是 IEEE 802.3 格式和 DIX 格式
4. （多选）下列描述中正确的是？（　）
   A. 以太帧中的目的 MAC 地址只能是一个单播 MAC 地址
   B. 以太帧中的源 MAC 地址只能是一个单播 MAC 地址
   C. 组播帧的源 MAC 地址一定是一个单播 MAC 地址

3.3　以太网交换机

如果交换机转发数据的端口都是以太网口，则这样的交换机称为以太网交换机（Ethernet Switch）；如果交换机转发数据的端口都是令牌环端口，则这样的交换机称为令牌环交换机（Token Ring Switch），如此等等。以太网交换机、令牌环交换机等都是局域网交换机（LAN Switch）这个家庭中的成员。从理论上讲，局域网交换机的成员有很多，但实际上，除了以太网交换机外，其他成员基本上已被市场机制所淘汰。所以，目前以太网交换机与局域网交换机几乎成为了同一个概念。本书所说的交换机，如无特别说明，都是指以太网交换机。

学习完本节内容之后，我们应该能够：

（1）熟悉交换机的 3 种转发操作；

（2）熟悉交换机对于单播帧和广播帧的转发原理和过程；

（3）熟悉交换机是如何学习 MAC 地址与端口的映射关系的；

（4）熟悉计算机的端口对于收到的单播帧和广播帧的处理过程；

（5）理解 MAC 地址表的老化机制。

3.3.1　3 种转发操作

交换机会对通过传输介质进入其端口的每一个帧都进行转发操作，交换机的基本作用就是用来转发帧的。

如图 3-7 所示，交换机对于从传输介质进入其某一端口的帧的转发操作一共有 3 种：泛洪（Flooding）、转发（Forwarding）、丢弃（Discarding）。

（1）泛洪：交换机把从某一端口进来的帧通过所有其他的端口转发出去（注意，“所有其他的端口”是指除了这个帧进入交换机的那个端口以外的所有端口）。泛洪操作是一种点到多点的转发行为。

（2）转发：交换机把从某一端口进来的帧通过另一个端口转发出去（注意，“另一个端口”不能是这个帧进入交换机的那个端口）。这里的转发操作是一种点到点的转发行为。

（3）丢弃：交换机把从某一端口进来的帧直接丢弃。丢弃操作其实就是不进行转发。

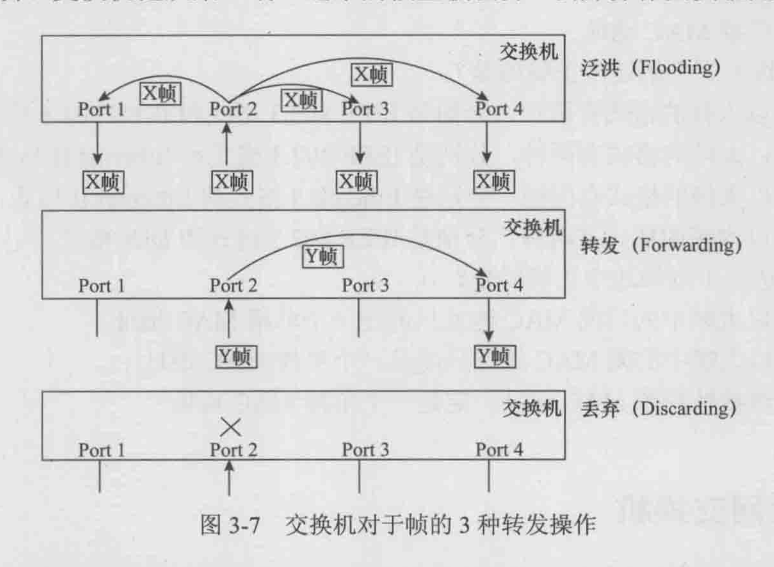


图 3-7　交换机对于帧的 3 种转发操作

图 3-7 中的箭头表示帧的运动轨迹。关于这些运动轨迹的细节描述请读者认真复习 3.1.2 小节的内容。

泛洪操作、转发操作、丢弃操作这 3 种转发行为经常被笼统地称为转发（即一般意义上的转发）操作，因此读者在遇到“转发”一词时，需要根据上下文搞清楚它究竟是一般意义上的转发呢，还是特指点到点转发的意思。

3.3.2　交换机的工作原理

交换机的工作原理主要是指交换机对于从传输介质进入其端口的帧进行转发的过程。在下面的描述中，将出现诸如“MAC 地址表”等一些读者可能觉得完全陌生或不太明白的概念。读者先不用着急，随着学习的继续和深入，自然会熟悉和理解这些概念。

每台交换机中都有一个 MAC 地址表，它存放了 MAC 地址与交换机端口编号之间的映射关系。MAC 地址表存在于交换机的工作内存中，交换机刚上电时，MAC 地址表中没有任何内容，是一个空表。随着交换机不断地转发数据并进行地址学习，MAC 地址表的内容会逐步丰富起来。当交换机下电或重启时，MAC 地址表的内容会完全丢失。

交换机的基本工作原理（转发原理）可以概括地描述如下。

（1）如果从传输介质进入交换机的某个端口的帧是一个单播帧，则交换机会去 MAC 地址表中查找这个帧的目的 MAC 地址。

1）如果查不到这个 MAC 地址，则交换机将对该帧执行泛洪操作。

2）如果查到了这个 MAC 地址，则比较这个 MAC 地址在 MAC 地址表中对应的端口编号是不是这个帧从传输介质进入交换机的那个端口的端口编号。

a）如果不是，则交换机将对该帧执行转发操作（将该帧送至该帧的目的 MAC 地址在 MAC 地址表中对应的那个端口，并从那个端口发送出去）。

b）如果是，则交换机将对该帧执行丢弃操作。

（2）如果从传输介质进入交换机的某个端口的帧是一个广播帧，则交换机不会去查 MAC 地址表，而是直接对该广播帧执行泛洪操作。

（3）如果从传输介质进入交换机的某个端口的帧是一个组播帧，则交换机的处理行为比较复杂，超出了本书的知识范围，这里略去不讲。

另外，交换机还具有 MAC 地址学习能力。当一个帧（无论是单播帧、组播帧，还是广播帧）从传输介质进入交换机后，交换机会检查这个帧的源 MAC 地址，并将该源 MAC 地址与这个帧进入交换机的那个端口的端口编号进行映射，然后将这个映射关系存放进 MAC 地址表。

以上是对交换机的转发原理的概括性描述，下面两小节将通过一些例子来展开对交换机转发原理的具体分析。

3.3.3　单交换机的数据转发示例

如图 3-8 所示，4 台计算机分别通过双绞线与同一台交换机相连。交换机有 4 个端口（Port），Port 后面的阿拉伯数字就是端口编号（Port No.），分别为 1，2，3，4。注意，双绞线两端所连接的其实分别是计算机上的网卡和交换机上的网卡（请复习 3.1 节的内容）。假设这 4 台计算机的网卡的 MAC 地址（即 BIA 地址）分别是 MAC1、MAC2、MAC3、MAC4；另外，假设交换机的 MAC 地址表此刻为空。

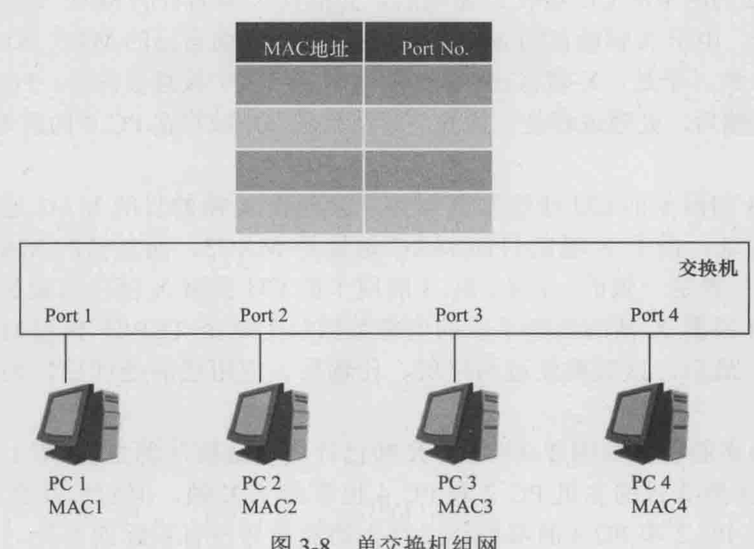


图 3-8　单交换机组网

现在，假设 PC 1 需要向 PC 3 发送一个单播帧 X（特别假设：PC 1 已经知道了 PC 3 的网卡的 MAC 地址为 MAC3），因此把 PC 1 称为源主机，PC 3 称为目的主机。下面的

步骤描述了 X 帧从 PC 1 运动到 PC 3 的全过程。

（1）PC 1 的应用软件所产生的数据经 TCP/IP 模型的应用层、传输层、网络层处理后，得到数据包（Packet）。数据包下传给 PC 1 的网卡的 CU 后，CU 会将之封装成帧。假设封装的第一个帧叫 X 帧，CU 会将 MAC3 作为 X 帧的目的 MAC 地址，然后会从自己的 ROM 中读出 BIA 地址（MAC1），并将 BIA 地址（MAC1）作为 X 帧的源地址。关于 X 帧的其他字段的内容我们暂不关心。至此，X 帧已在 PC 1 的网卡的 CU 中形成。

（2）X 帧接下来的运动轨迹如下：PC 1 的网卡的 CU → PC 1 的网卡的 OB → PC 1 的网卡的 LC → PC 1 的网卡的 TX → 双绞线 → Port 1 的网卡的 RX → Port 1 的网卡的 LD → Port 1 的网卡的 IB → Port 1 的网卡的 CU。这一过程在 3.1 节中有详细的说明，这里不再赘述。

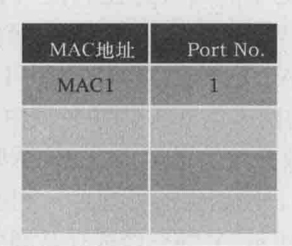
（3）X 帧到达 Port 1 的网卡的 CU 后，交换机会去 MAC 地址表中查找 X 帧的目的 MAC 地址 MAC3。由于此时 MAC 地址表是空表，所以在 MAC 地址表中查不到 MAC3。根据交换机的转发原理，交换机会对 X 帧执行泛洪操作。然后，交换机还要进行地址学习：因为 X 帧是从 Port 1 进入交换机的，并且 X 帧的源 MAC 地址为 MAC1，所以，交换机会将 MAC1 映射到 Port 1，并将这一映射关系作为一个条目写进 MAC 地址表。

（4）X 帧被执行泛洪操作后，Port *i*（*i* = 2，3，4）的网卡的 CU 都会从 Port 1 的网卡的 CU 那里获得一个 X 帧的拷贝。然后，这些拷贝的运动过程如下：Port *i* 的网卡的 CU→Port *i* 的网卡的 OB→Port *i* 的网卡的 LC→Port *i* 的网卡的 TX→双绞线→PC *i* 的网卡的 RX→PC *i* 的网卡的 LD→PC*i* 的网卡的 IB→PC *i* 的网卡的 CU。这一过程在 3.1 节中有详细的说明，这里不再赘述。

（5）PC 2 的网卡的 CU 在收到 X 帧后，会检查 X 帧的目的 MAC 地址是不是自己的 MAC 地址。由于 X 帧的目的 MAC 地址是 MAC3，而自己的 MAC 地址是 MAC2，所以二者不一致。于是，X 帧将在 PC 2 的网卡的 CU 中被直接丢弃。PC 4 的网卡的 CU 在收到 X 帧后，处理过程是一样的，其结果是，X 帧将在 PC 4 的网卡的 CU 中被直接丢弃。

（6）PC 3 的网卡的 CU 在收到 X 帧后，会检查 X 帧的目的 MAC 地址是不是自己的 MAC 地址。由于 X 帧的目的 MAC 地址是 MAC3，而自己的 MAC 地址也是 MAC3，所以二者是一致的。于是，PC 3 的网卡的 CU 会将 X 帧中的数据包（Packet）抽取出来，并根据 X 帧的类型字段的值将数据包上送至 TCP/IP 模型的网络层的相应处理模块。最后，该数据经过网络层、传输层、应用层的处理后，到达相应的应用软件。

至此，网络的状态如图 3-9 所示。X 帧已经成功地被从源主机 PC 1 送达至目的主机 PC 3，虽然非目的主机 PC 2 和 PC 4 也收到了 X 帧，但它们都会将 X 帧直接丢弃。X 帧在 PC 2 和 PC 4 的双绞线上产生的流量并没有实际的用处，这样的流量被称为垃圾流量。显然，这里的垃圾流量是因为交换机对 X 帧执行了泛洪操作而引起的。



MAC地址	Port No.
MAC1	1

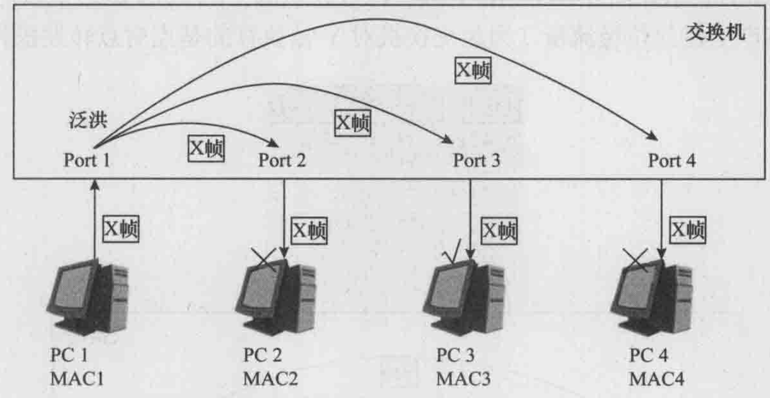


图 3-9　PC 1 向 PC 3 发送一个单播帧

现在，在图 3-9 所示的网络状态下，假设 PC 4 需要向 PC 1 发送一个单播帧 Y（特别假设：PC 4 已经知道了 PC 1 的网卡的 MAC 地址为 MAC1）。此时，PC 4 为源主机，PC 1 为目的主机。下面的步骤描述了 Y 帧从 PC 4 运动到 PC 1 的全过程。

（1）PC 4 的应用软件所产生的数据经 TCP/IP 模型的应用层、传输层、网络层处理后，得到数据包（Packet）。数据包下传给 PC 4 的网卡的 CU 后，CU 会将之封装成帧。假设封装的第一个帧叫 Y 帧，CU 会将 MAC1 作为 Y 帧的目的 MAC 地址，然后会从自己的 ROM 中读出 BIA 地址（MAC4），并将 BIA 地址（MAC4）作为 Y 帧的源地址。关于 Y 帧的其他字段的内容我们暂不关心。至此，Y 帧已在 PC 4 的网卡的 CU 中形成。

（2）Y 帧接下来的运动轨迹如下：PC 4 的网卡的 CU→PC 4 的网卡的 OB→PC 4 的网卡的 LC→PC 4 的网卡的 TX→双绞线→Port 4 的网卡的 RX→Port 4 的网卡的 LD→Port 4 的网卡的 IB→Port 4 的网卡的 CU。

（3）Y 帧到达 Port 4 的网卡的 CU 后，交换机会去 MAC 地址表中查找 Y 帧的目的 MAC 地址 MAC1。查表的结果是，MAC1 对应了 Port 1，而 Port 1 不是 Y 帧的入端口 Port 4。根据交换机的转发原理，交换机会对 Y 帧执行点到点转发操作，也就是将 Y 帧送至 Port 1 的网卡的 CU。然后，交换机还要进行地址学习：因为 Y 帧是从 Port 4 进入交换机的，并且 Y 帧的源 MAC 地址为 MAC4，所以，交换机会将 MAC4 映射到 Port 4，并将这一映射关系作为一个新的条目写进 MAC 地址表。

（4）Y 帧到达 Port 1 的网卡的 CU 后，接下来的运动过程如下：Port 1 的网卡的 CU→Port 1 的网卡的 OB→Port 1 的网卡的 LC→Port 1 的网卡的 TX→双绞线→PC 1 的网

卡的 RX→PC 1 的网卡的 LD→PC 1 的网卡的 IB→PC 1 的网卡的 CU。

（5）PC 1 的网卡的 CU 在收到 Y 帧后，会检查 Y 帧的目的 MAC 地址是不是自己的 MAC 地址。由于 Y 帧的目的 MAC 地址是 MAC1，而自己的 MAC 地址也是 MAC1，所以二者是一致的。于是，PC 1 的网卡的 CU 会将 Y 帧中的数据包（Packet）抽取出来，并根据 Y 帧的类型字段的值将数据包上送至 TCP/IP 模型的网络层的相应处理模块。最后，该数据经过网络层、传输层、应用层的处理后，到达相应的应用软件。

至此，网络的状态如图 3-10 所示。Y 帧已经成功从源主机 PC 4 送达至目的主机 PC 1，并且这次没有产生任何垃圾流量（因为交换机对 Y 帧执行的是点对点转发操作）。

MAC地址	Port No.
MAC1	1
MAC4	4

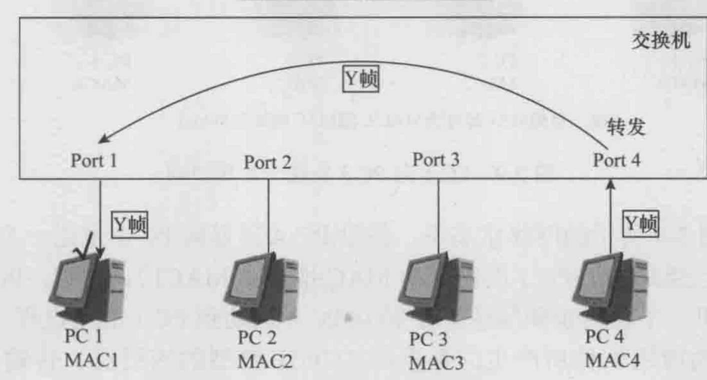


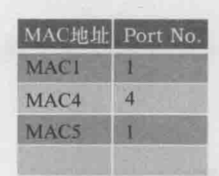
图 3-10 PC 4 向 PC 1 发送一个单播帧

现在，在图 3-10 所示的网络状态下，假设 PC 1 将发送一个单播帧 Z。由于某种未知的原因（比如由于 Bug 的原因），在 PC 1 的网卡的 CU 中形成的 Z 帧的目的 MAC 地址为 MAC1，源 MAC 地址为 MAC5。下面的步骤描述了 Z 帧的运动轨迹。

（1）PC 1 的网卡的 CU→PC 1 的网卡的 OB→PC 1 的网卡的 LC→PC 1 的网卡的 TX→双绞线→Port 1 的网卡的 RX→Port 1 的网卡的 LD→Port 1 的网卡的 IB→Port 1 的网卡的 CU。

（2）Z 帧到达 Port 1 的网卡的 CU 后，交换机会去 MAC 地址表中查找 Z 帧的目的 MAC 地址 MAC1。查表的结果是，MAC1 对应了 Port 1，而 Port 1 正是 Z 帧的入端口。根据交换机的转发原理，交换机会对 Z 帧执行丢弃操作。然后，交换机还要进行地址学习：因为 Z 帧是从 Port 1 进入交换机的，并且 Z 帧的源 MAC 地址为 MAC5，所以，交换机会将 MAC5 映射到 Port 1，并将这一映射关系作为一个新的条目写进 MAC 地址表。

至此，网络的状态如图 3-11 所示。



MAC地址	Port No.
MAC1	1
MAC4	4
MAC5	1

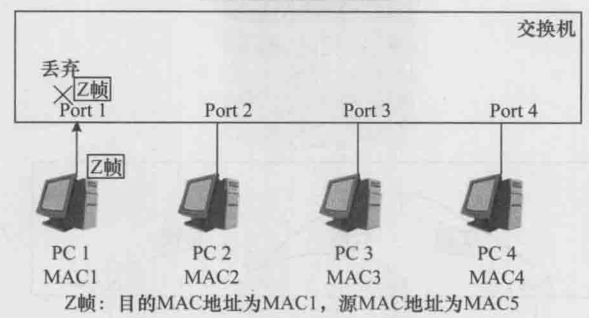


图 3-11　PC 1 发送一个单播帧

图 3-9、图 3-10、图 3-11 分别说明了交换机对计算机发送的单播帧执行泛洪操作、转发操作、丢弃操作的过程。现在，再来看看计算机发送广播帧的例子。假定目前的网络状态如图 3-11 所示，而 PC 3 将要发送一个广播帧 W。下面的步骤描述了 W 帧的运动轨迹。

（1）PC 3 希望把应用软件所产生的数据同时发送给所有其他的计算机。这些数据经 TCP/IP 模型的应用层、传输层、网络层处理后，得到数据包（Packet）。数据包下传给 PC 3 的网卡的 CU 后，CU 会将之封装成广播帧。假设封装的第一个帧叫 W 帧，CU 会将广播地址作为 W 帧的目的 MAC 地址，然后会从自己的 ROM 中读出 BIA 地址（MAC3），并将 BIA 地址（MAC3）作为 W 帧的源地址。关于 W 帧的其他字段的内容我们暂不关心。至此，W 帧已在 PC 3 的网卡的 CU 中形成。

（2）W 帧接下来的运动轨迹如下：PC 3 的网卡的 CU→PC 3 的网卡的 OB→PC 3 的网卡的 LC→PC 3 的网卡的 TX→双绞线→Port 3 的网卡的 RX→Port 3 的网卡的 LD→Port 3 的网卡的 IB→Port 3 的网卡的 CU。

（3）W 帧到达 Port 3 的网卡的 CU 后，交换机不会去查 MAC 地址表，而是直接对 W 帧执行泛洪操作，这是因为交换机能判断出 W 帧是一个广播帧。然后，交换机还要进行地址学习：因为 W 帧是从 Port 3 进入交换机的，并且 W 帧的源 MAC 地址为 MAC3，所以，交换机会将 MAC3 映射到 Port 3，并将这一映射关系作为一个新的条目写进 MAC 地址表。

（4）W 帧被执行泛洪操作后，Port *i*（*i* =1，2，4）的网卡的 CU 都会从 Port 3 的网卡的 CU 那里获得一个 W 帧的拷贝。然后，这些拷贝的运动过程如下：Port *i* 的网卡的 CU→Port *i* 的网卡的 OB→Port *i* 的网卡的 LC→Port *i* 的网卡的 TX→双绞线→PC *i* 的网卡的 RX→PC *i* 的网卡的 LD→PC *i* 的网卡的 IB→PC *i* 的网卡的 CU。

（5）PC *i*（*i* = 1，2，4）的网卡的 CU 在收到 W 帧后，判断出 W 帧是一个广播帧，于是会将 W 帧中的数据包（Packet）抽取出来，并根据 W 帧的类型字段的值将数据包上送至 TCP/IP 模型的网络层的相应处理模块。最后，该数据经过网络层、传输层、应用

层的处理后，到达相应的应用软件。

至此，PC 1、PC 2、PC 4 的应用软件都收到了同样的、来自 PC 3 的应用软件的数据，网络的状态如图 3-12 所示。



MAC地址	Port No.
MAC1	1
MAC4	4
MAC5	1
MAC3	3

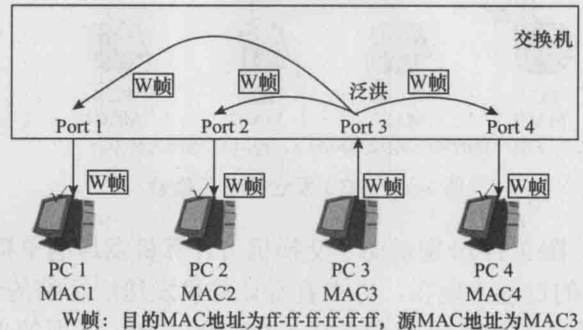


图 3-12 PC 3 发送一个广播帧

通过前面这些例子，我们详细地描述了交换机是如何对单播帧和广播帧进行转发的。关于组播帧的情况，我们不作描述，因为这已超出了本书的知识范围。

作为本小节的结束，我们还需要强调一个重要的知识点，内容如下。

（1）当计算机的网卡收到一个单播帧时，会将该单播帧的目的 MAC 地址与自己的 MAC 地址进行比较。如果二者相同，则网卡会根据该单播帧的类型字段的值将该单播中的载荷数据上送至网络层中的相应处理模块。如果二者不同，则网卡会将该单播帧直接丢弃。

（2）当计算机的网卡收到一个广播帧时，会直接根据该广播帧的类型字段的值将该广播中的载荷数据上送至网络层中的相应处理模块。

（3）当交换机的网卡收到一个单播帧时，不会将该单播帧的目的 MAC 地址与自己的 MAC 地址进行比较，而是直接去查 MAC 地址表，并根据查表的结果决定对该单播帧执行 3 种转发操作的哪一种。

（4）当交换机的网卡收到一个广播帧时，直接对该广播帧执行泛洪操作。

3.3.4 多交换机的数据转发示例

如图 3-13 所示，3 台交换机通过双绞线与 4 台计算机相连，形成了一个相对复杂的网络。假设交换机的 MAC 地址表此刻都为空。下面，我们就通过一些例子来说明帧在这个网络中的转发过程。由于上一小节已经对交换机的转发原理进行了详细的展示，所以下面的描述相对比较简洁。如果读者有不明白的地方，则应认真复习一下上一小节的内容。

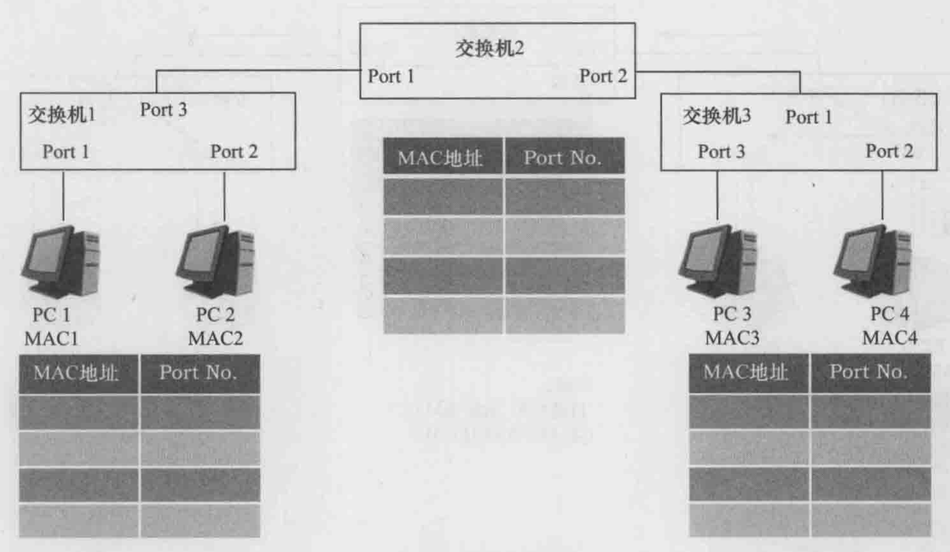


图 3-13 多交换机组网

现在，假设 PC 1 需要向 PC 3 发送一个单播帧 X（特别假设：PC 1 已经知道了 PC 3 的网卡的 MAC 地址为 MAC3）。下面的步骤描述了 X 帧从 PC 1 运动到 PC 3 的全过程。

（1）PC 1 的 CU 完成 X 帧的封装；X 帧的目的 MAC 地址为 MAC3，源 MAC 地址为 MAC1。

（2）X 帧接下来的运动轨迹如下：PC 1 的网卡的 CU → 交换机 1 的 Port 1 的网卡的 CU。

（3）交换机 1 对 Port 1 的网卡的 CU 中的 X 帧执行泛洪操作，一路通过交换机 1 的 Port 3 到达交换机 2 的 Port 1，另一路通过交换机 1 的 Port 2 到达 PC 2。交换机 1 将 MAC1 与 Port 1 的对应关系写进自己的 MAC 地址表。

（4）交换机 2 对 Port 1 的网卡的 CU 中的 X 帧执行泛洪操作，X 帧通过交换机 2 的 Port 2 到达交换机 3 的 Port 1。交换机 2 将 MAC1 与 Port 1 的对应关系写进自己的 MAC 地址表。

（5）交换机 3 对 Port 1 的网卡的 CU 中的 X 帧执行泛洪操作，一路通过交换机 3 的 Port 3 到达 PC 3，另一路通过交换机 3 的 Port 2 到达 PC 4。交换机 3 将 MAC1 与 Port 1 的对应关系写进自己的 MAC 地址表。

（6）PC 2 和 PC 4 的网卡会将收到的 X 帧丢弃。PC 3 会将 X 帧的载荷数据上送给网络层。

至此，网络的状态如图 3-14 所示。X 帧已经成功从源主机 PC 1 送达至目的主机 PC 3，虽然非目的主机 PC 2 和 PC 4 也收到了 X 帧，但它们都会将 X 帧直接丢弃。

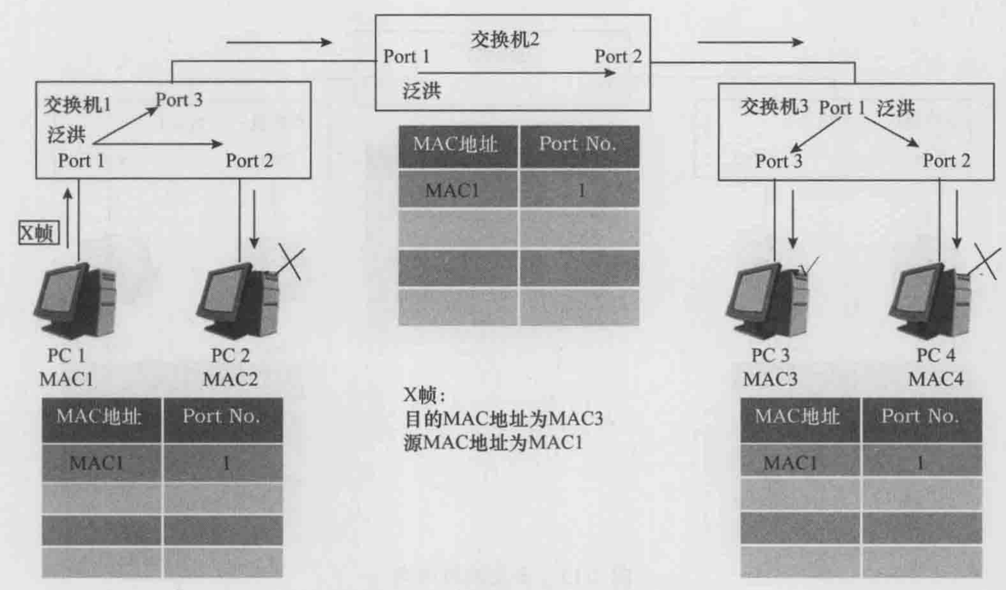


图 3-14 PC 1 向 PC 3 发送一个单播帧

现在，在图 3-14 所示的网络状态下，假设 PC 4 需要向 PC 1 发送一个单播帧 Y（特别假设：PC 4 已经知道了 PC 1 的网卡的 MAC 地址为 MAC1）。下面的步骤描述了 Y 帧从 PC 4 运动到 PC 1 的全过程。

（1）PC 4 的 CU 完成 Y 帧的封装；Y 帧的目的 MAC 地址为 MAC1，源 MAC 地址为 MAC4。

（2）Y 帧接下来的运动轨迹如下：PC 4 的网卡的 CU → 交换机 3 的 Port 2 的网卡的 CU。

（3）交换机 3 对 Port 2 的网卡的 CU 中的 Y 帧执行点对点转发操作，Y 帧通过交换机 3 的 Port 1 到达交换机 2 的 Port 2。交换机 3 将 MAC4 与 Port 2 的对应关系写进自己的 MAC 地址表。

（4）交换机 2 对 Port 2 的网卡的 CU 中的 Y 帧执行点对点转发操作，Y 帧通过交换机 2 的 Port 1 到达交换机 1 的 Port 3。交换机 2 将 MAC4 与 Port 2 的对应关系写进自己的 MAC 地址表。

（5）交换机 1 对 Port 3 的网卡的 CU 中的 Y 帧执行点对点转发操作，Y 帧通过交换机 1 的 Port 1 到达 PC 1。交换机 1 将 MAC4 与 Port 3 的对应关系写进自己的 MAC 地址表。

（6）PC 1 会将收到的 Y 帧的载荷数据上送给网络层。

至此，网络的状态如图 3-15 所示。Y 帧已经成功从源主机 PC 4 送达至目的主机 PC 1，并且这次没有产生任何垃圾流量。

接下来，我们假定网络的状态如图 3-16 所示，且 PC 2 需要向 PC 1 发送一个单播帧 Z（特别假设：PC 2 已经知道了 PC 1 的网卡的 MAC 地址为 MAC1）。注意，此时交换机 1 的 MAC 地址表中并没有关于 MAC1 的条目，但交换机 2 的 MAC 地址表中存在关

于 MAC1 的条目。

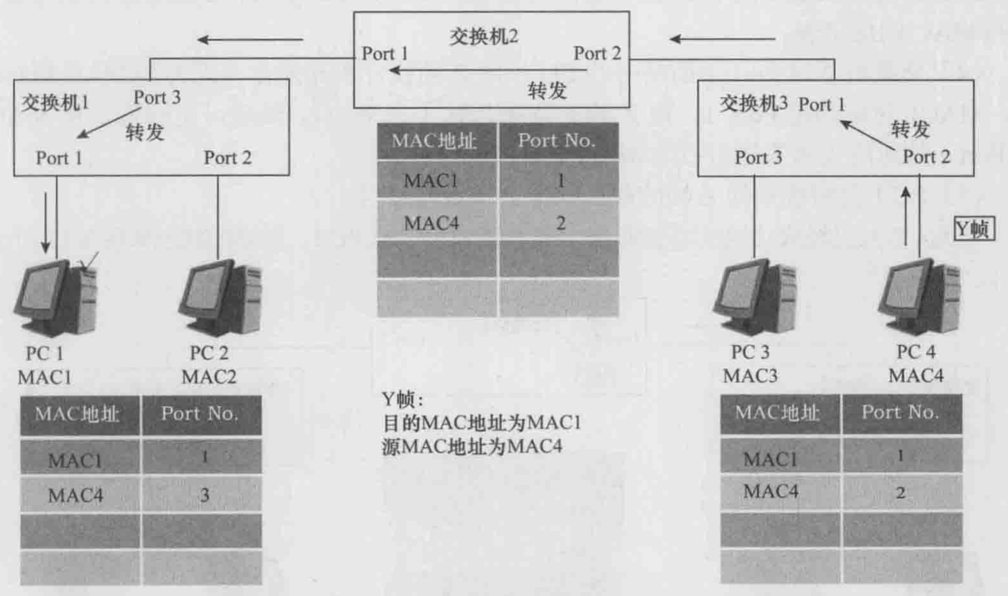


图 3-15 PC 4 向 PC 1 发送一个单播帧

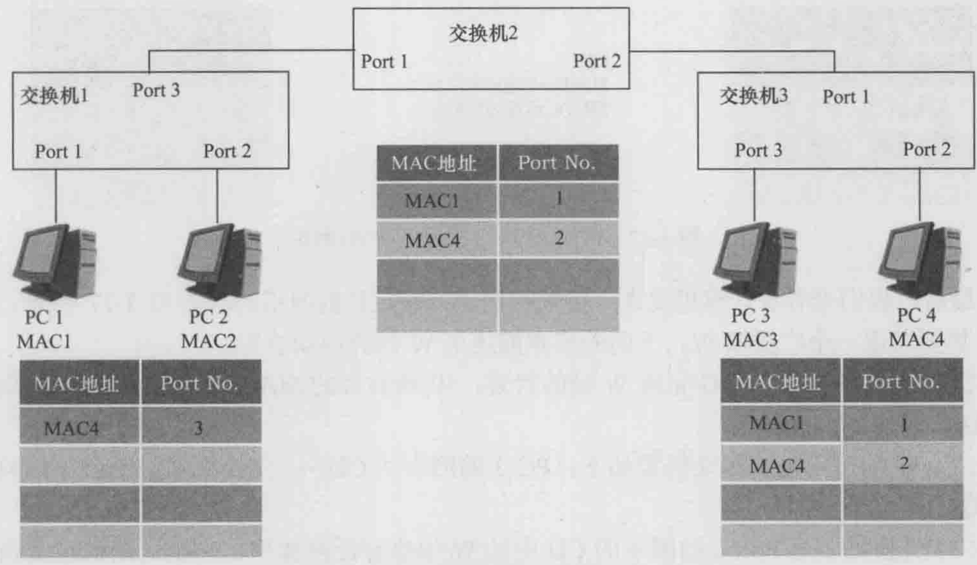


图 3-16 PC 2 需要向 PC 1 发送一个单播帧

下面的步骤描述了 Z 帧从 PC 2 运动到 PC 1 的全过程。

（1）PC 2 的网卡的 CU 完成 Z 帧的封装；Z 帧的目的 MAC 地址为 MAC1，源 MAC 地址为 MAC2。

（2）Z 帧接下来的运动轨迹如下：PC 2 的网卡的 CU → 交换机 1 的 Port 2 的网卡的 CU。

（3）交换机 1 对 Port 2 的网卡的 CU 中的 Z 帧执行泛洪操作（因为此时 MAC 地址

表中查不到 MAC1）。Z 帧一路通过交换机 1 的 Port 1 到达 PC 1，另一路通过交换机 1 的 Port 3 到达交换机 2 的 Port 1。然后，交换机 1 将 MAC2 与 Port 2 的对应关系写进自己的 MAC 的地址表。

（4）交换机 2 对 Port 1 的网卡的 CU 中的 Z 帧执行丢弃操作（因为在 MAC 地址表中，MAC1 对应的是 Port 1，而 Z 帧正是从 Port 1 进来的）。然后，交换机 2 将 MAC2 与 Port 1 的对应关系写进自己的 MAC 地址表。

（5）PC 1 会将收到的 Z 帧的载荷数据上送给网络层。

至此，Z 帧已经成功地从源主机 PC 2 运动到目的主机 PC 1，网络的状态如图 3-17 所示。

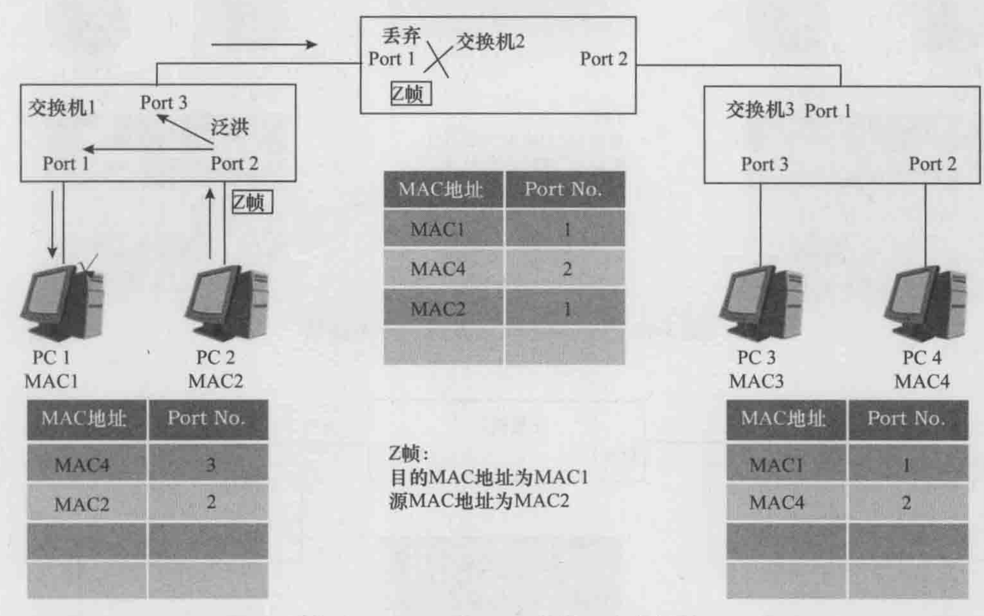


图 3-17　PC 2 向 PC 1 发送一个单播帧

最后，我们来看看计算机发送广播帧的例子。假定目前网络的状态如 3-17 所示，而 PC 3 将要发送一个广播帧 W。下面的步骤描述了 W 帧的运动轨迹。

（1）PC 3 的网卡的 CU 完成 W 帧的封装；W 帧的目的 MAC 地址为 ff-ff-ff-ff-ff-ff，源 MAC 地址为 MAC3。

（2）W 帧接下来的运动轨迹如下：PC 3 的网卡的 CU → 交换机 3 的 Port 3 的网卡的 CU。

（3）交换机 3 对 Port 3 的网卡的 CU 中的 W 帧执行泛洪操作，W 帧一路通过交换机 3 的 Port 1 到达交换机 2 的 Port 2，另一路通过交换机 3 的 Port 2 到达 PC 4。然后，交换机 3 将 MAC3 与 Port 3 的对应关系写进自己的 MAC 地址表。

（4）交换机 2 对 Port 2 的网卡的 CU 中的 W 帧执行泛洪操作，W 帧通过交换机 2 的 Port 1 到达交换机 1 的 Port 3。然后，交换机 2 将 MAC3 与 Port 2 的对应关系写进自己的 MAC 地址表。

（5）交换机 1 对 Port 3 的网卡的 CU 中的 W 帧执行泛洪操作，W 帧通过交换机 1 的 Port 1 和 Port 2 分别到达 PC 1 和 PC 2。然后，交换机 1 将 MAC3 与 Port 3 的对应关

系写进自己的 MAC 地址表。

（6）PC 4、PC 2、PC 1 的网卡在收到 W 帧后，会将 W 帧的载荷数据上送到网络层。

至此，PC 1、PC 2、PC 4 都接收到了来自 PC 3 的广播帧 W，网络的状态如图 3-18 所示。

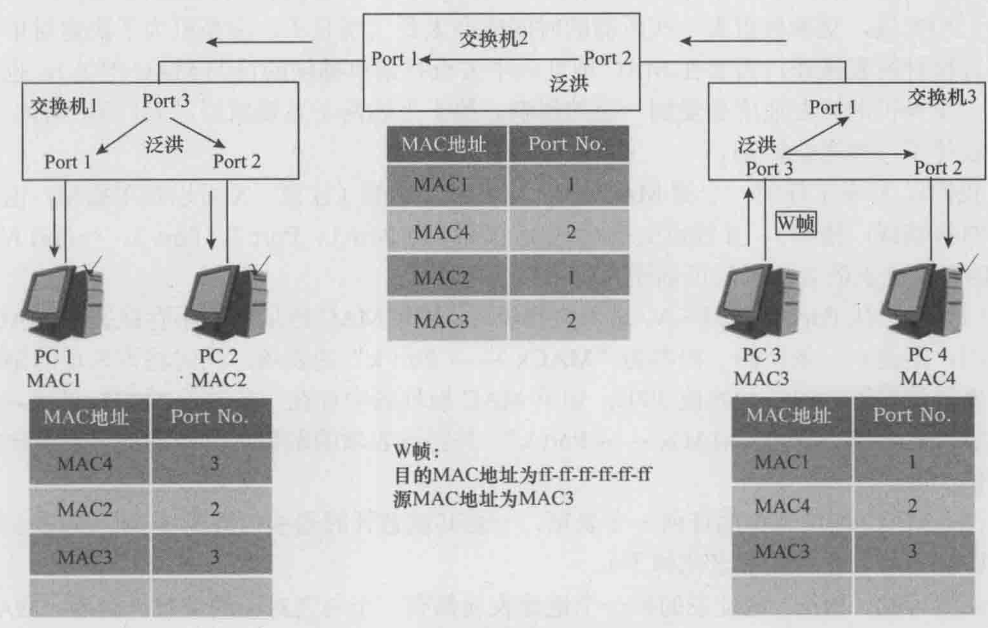


图 3-18　PC 3 发送一个广播帧

前面这些例子描述了帧在图 3-13 所示的网络中的运动情况。如果真的理解了这些例子，那么对于更为复杂的网络（见图 3-19），自然也会清楚帧在其中的运动过程。

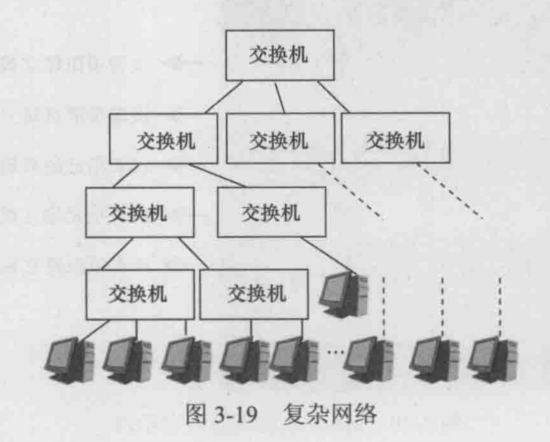


图 3-19　复杂网络

3.3.5　MAC 地址表

交换机的 MAC 地址表也称为 MAC 地址映射表，其中的每一个条目也称为一个地址表项，地址表项反映了 MAC 地址与端口的映射关系。前面的两个小节已经描述了交换机是如何学习 MAC 地址与端口的映射关系，并将这种映射关系作为地址表项写进

MAC 地址表的。

在现实中，交换机或计算机在网络中的位置可能会发生变化。如果交换机或计算机的位置真的发生了变化，那么交换机的 MAC 地址表中某些原来的地址表项很可能会错误地反映当前 MAC 地址与端口的映射关系。另外，MAC 地址表中的地址表项如果太多，那么平均来说，交换机查表一次所需的时间就会太长（别忘了，交换机为了决定对单播帧执行何种转发操作，需要在 MAC 地址表中去查找该单播帧的目的 MAC 地址），也就是说，交换机的转发速度会受到一定的影响。鉴于上述两个主要原因，人们为 MAC 地址表设计了一种老化机制。

我们以 X 表示任何一个源 MAC 地址为 MACx 的帧（注意，X 可以是单播帧，也可以是组播帧或广播帧），并假设交换机的 N 个端口为 Port 1、Port 2、Port 3、…Port N，则 MAC 地址表的老化机制可描述为以下两条原则。

（1）当 X 从 Port k（$1 \leqslant k \leqslant N$）进入交换机时，如果 MAC 地址表中不存在关于 MACx 的表项，则建立一条新的、内容为“MACx ←→ Port k”的表项，同时将该表项的倒数计时器的值设置为缺省初始值 300s；如果 MAC 地址表中存在一条关于 MACx 的表项，则该表项的内容更新为“MACx ←→ Port k”，并将该表项的倒数计时器的值重置为缺省初始值 300s。

（2）MAC 地址表中的任何一个表项，一旦其倒数计时器的值降为 0 时，则该表项将立即被删除（也就是被老化掉了）。

从上可知，MAC 地址表的每一个地址表项都有一个与之对应的倒数计时器。MAC 地址表的内容是动态的，新的表项不断被建立，老的表项不断被更新或被删除。图 3-20 显示了某一时刻某台交换机的 MAC 地址表的内容。

MAC地址	Port No.	倒数计时器（秒）	
00-1e-10-00-00-02	3	240	→ 该表项距建立或最近一次刷新已有60s
00-1e-10-00-0d-07	5	300	→ 该表项刚被建立或刷新
			→ 该表项已经被删除（老化掉）
00-1e-10-00-00-a8	2	95	→ 该表项距建立或最近一次刷新已有205s
00-1e-10-00-00-05	1	10	→ 该表项距建立或最近一次刷新已有290s
……	……	……	
……	……	……	

图 3-20 MAC 地址表的内容示例

显然，倒数计时器的初始值越小，MAC 地址表的动态性就越强。标准规定的倒数计时器的缺省初始值为 300s，但这个初始值通常是可以通过配置命令进行修改的。读者可以去思考一下，倒数计时器的初始值太大（比如 3 天）或太小（比如 1s）的后果是什么？

顺便提一下，在现实中，一台低档的交换机的 MAC 地址表通常最多可以存放数千

条地址表项；一台中档的交换机的 MAC 地址表通常最多可以存放数万条地址表项；一台高档的交换机的 MAC 地址表通常最多可以存放几十万条地址表项。

3.3.6　练习题

1.（单选）一台交换机有 8 个端口，一个单播帧从某一端口进入了该交换机，但交换机在 MAC 地址表中查不到关于该帧的目的 MAC 地址的表项，那么交换机对该帧进行的转发操作是？（　）

A．丢弃　　B．泛洪　　C．点对点转发

2.（单选）一台交换机有 8 个端口，一个单播帧从某一端口进入了该交换机，交换机在 MAC 地址表中查到了关于该帧的目的 MAC 地址的表项，那么交换机对该帧进行的转发操作是？（　）

A．一定是点对点转发　　B．一定是丢弃

C．可能是点对点转发，也可能是丢弃　D．泛洪

3.（多选）下列描述中正确的是？（　）

A．计算机的端口在收到一个广播帧后，一定会将帧中的载荷数据送给上层协议去处理

B．计算机的端口在收到一个广播帧后，会对该帧执行泛洪操作

C．交换机的某个端口在收到一个广播帧后，一定会将帧中的载荷数据送给上层协议去处理

D．交换机的某个端口在收到一个广播帧后，会对该帧执行泛洪操作

4.（多选下列描述中正确的是？（　）

A．计算机中的 MAC 地址表也具有老化机制

B．从统计的角度看，如果交换机 MAC 地址表中的地址表项越少，则交换机执行泛洪操作的可能性就越大

C．从统计的角度看，如果交换机 MAC 地址表中的地址表项越少，则网络中出现垃圾流量的可能性就越大

5.（单选）标准规定，MAC 地址表中的倒数计时器的缺省初始值是？（　）

A．100s　　B．5min　　C．30min

3.4　ARP

ARP（Address Resolution Protocol）是一个非常重要并经常使用的地址解析协议，它虽然是一个网络层协议，但却涉及一些数据链路层的信息。ARP 协议的基本作用是根据已知的 IP 地址获得其对应的 MAC 地址。

学习完本节内容之后，我们应该能够：

（1）理解 ARP 协议的工作原理；

（2）理解 ARP 报文中各个字段的含义和作用；

（3）理解 ARP 缓存表的作用。

3.4.1 ARP 的基本原理

在 3.3 节中，我们分析帧在交换网络中的运动过程时，总是特别假定源计算机已经知道了目的计算机的 MAC 地址。事实上，一个源设备开始是不知道目的设备的 MAC 地址的。源设备总是通过某种机制（例如 DNS，Domain Name System）先获取到目的设备的 IP 地址（后续的章节会专门讲解有关 IP 地址的知识），然后利用 ARP 协议对此 IP 地址进行解析，从而获取到目的设备的 MAC 地址。当然，一个设备总是知道自己的 MAC 地址和 IP 地址的。

源设备需要解析一个 IP 地址时，会发出一个广播帧，广播帧的载荷数据是一个 ARP 请求报文。目的设备在接收到 ARP 请求报文后，会向源设备发送一个单播帧，该单播帧的载荷数据是一个 ARP 应答报文，该 ARP 应答报文中包含了目的设备的 MAC 地址。

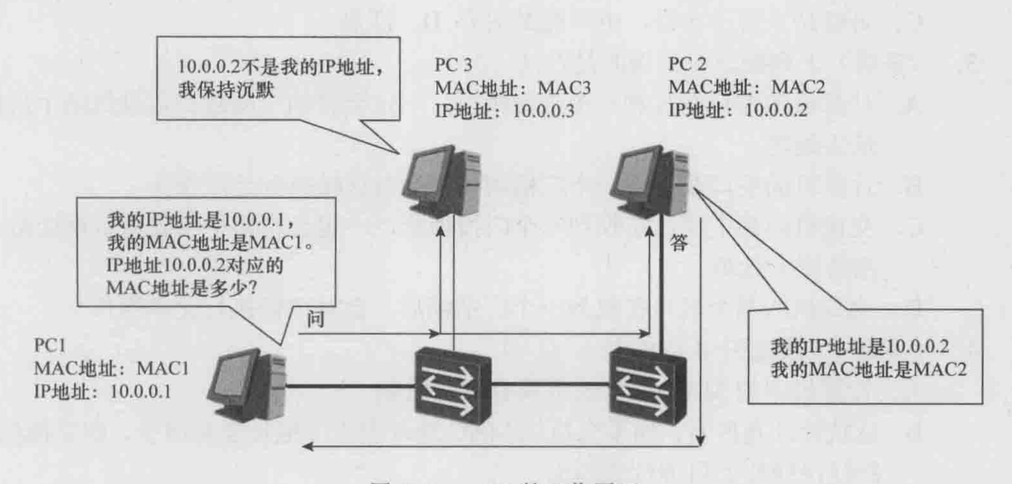


图 3-21 ARP 的工作原理

下面通过一个例子来说明 ARP 的基本工作原理。

如图 3-21 所示，假设源主机 PC 1 已经知道了目的主机 PC 2 的 IP 地址为 10.0.0.2，现在需要获取 PC 2 的 MAC 地址。PC 1 获取 PC 2 的 MAC 地址的过程如下。

（1）PC 1 会发送一个广播帧，该广播帧的源 MAC 地址为 MAC 1，类型字段的值是 0x0806，表明该广播帧的载荷数据是一个 ARP 报文（具体为 ARP 请求报文）。该 ARP 请求报文的含义是：我的 IP 地址是 10.0.0.1，我的 MAC 地址是 MAC1，请问 IP 地址 10.0.0.2 所对应的 MAC 地址是多少？

（2）因为 PC 1 发送的是一个广播帧，所以 PC 2 和 PC 3 都会接收到它，并根据其类型字段的值（0x0806）将其中的 ARP 请求报文上送给网络层的 ARP 处理模块进行处理。

（3）PC 3 的 ARP 处理模块会发现，10.0.0.2 并不是自己的 IP 地址，所以不会进行应答，而是将 ARP 请求报文中 10.0.0.1 与 MAC1 的对应关系存放进自己的 ARP 缓存表，然后将此 ARP 请求报文丢弃。

（4）PC 2 的 ARP 处理模块会发现，10.0.0.2 正是自己的 IP 地址，所以会进行应答。

PC 2会向PC 1发送一个单播帧，该帧的目的MAC地址为MAC1，源MAC地址为MAC2，类型字段的值还是0x0806。该帧的载荷数据是一个ARP应答报文，应答报文中包含了PC 2的IP地址10.0.0.2和MAC地址MAC2。另外，PC 2也要将所收到的ARP请求报文中的10.0.0.1与MAC1的对应关系存放进自己的ARP缓存表。

（5）PC 1在收到PC 2发送的单播帧后，会将其中的ARP应答报文上送给三层的ARP处理模块。PC 1的ARP处理模块会从该应答报文中获取到PC 2的MAC地址MAC2。另外，PC 1也会将10.0.0.2与MAC2的对应关系存放进自己的ARP缓存表。

从上面的描述中，我们接触到了ARP缓存表这个概念。设备中的ARP缓存表是用来临时存放IP地址与MAC地址的对应关系的。当某一设备需要向目的设备发送单播帧时，会首先查看自己的ARP缓存表中是否已经有了目的设备的MAC地址。如果有，就直接使用它；如果没有，就会发起ARP请求来获取它。

ARP缓存表也具有动态特性：一个条目（即一个IP地址与MAC地址的对应关系）从其被建立或最近一次被使用算起，会有180s（该时间值可通过配置进行修改）的生存期；一旦过了生存期，该条目就会被删除。某个条目在每次被使用时，该条目的生存期都会被重新设置为180s。

3.4.2 ARP的报文格式

ARP报文分为ARP请求报文和ARP应答报文，这两种报文的结构相同，但是各个字段的取值有所不同，具体结构如图3-22所示（有阴影的区域才是ARP报文）。

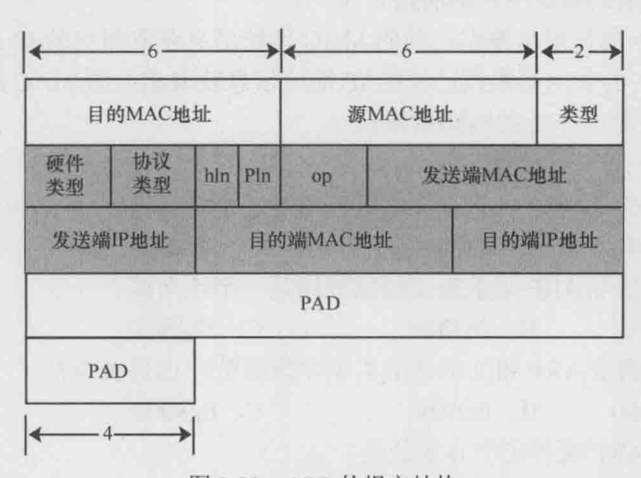


图3-22　ARP的报文结构

ARP报文中各个字段的含义如表3-1所示。

表3-1　ARP报文中各个字段的含义

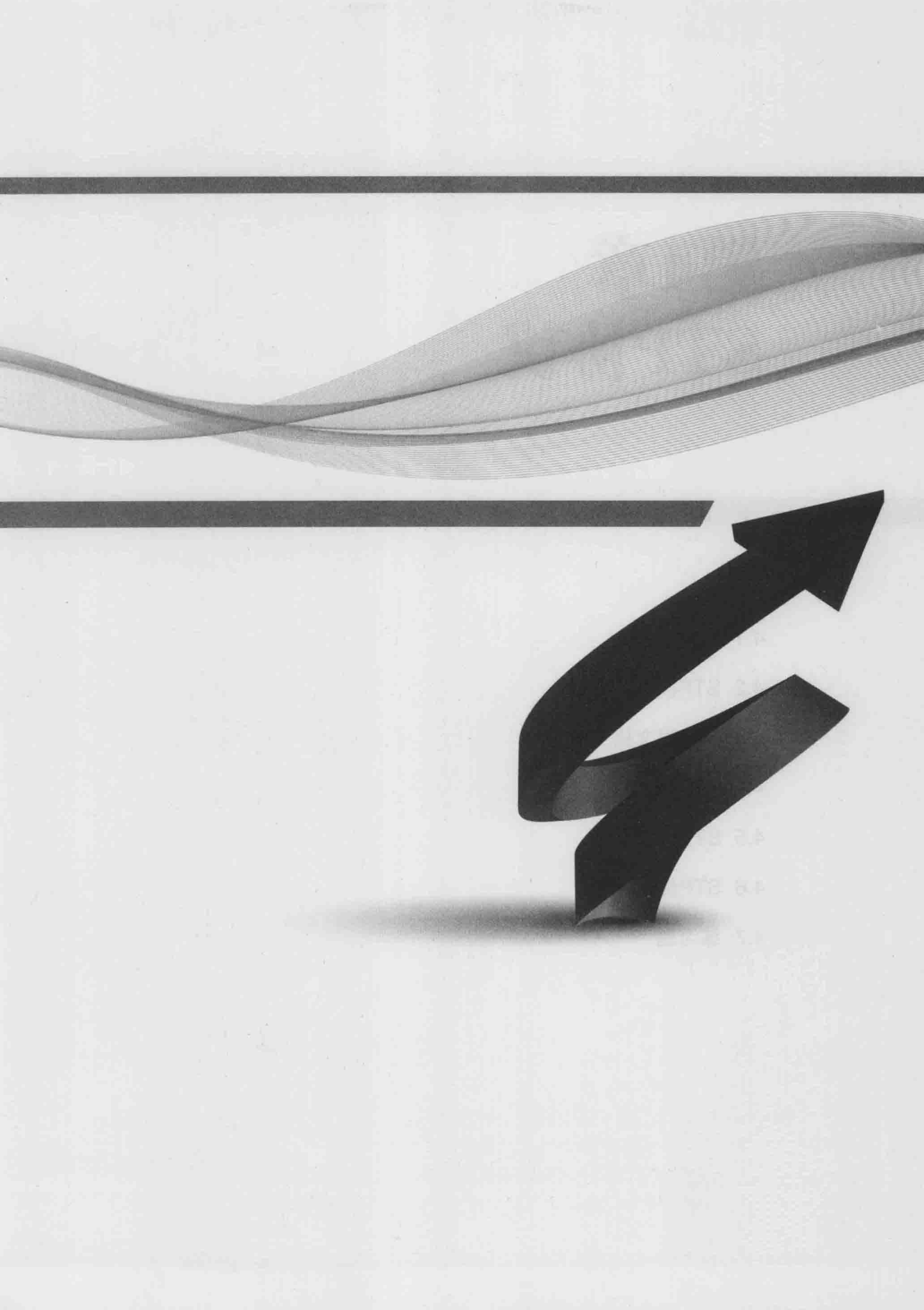
字段	ARP请求报文	ARP应答报文
目的MAC地址	ff-ff-ff-ff-ff-ff	请求端的MAC地址
源MAC地址	请求端的MAC地址	被请求端的MAC地址
类型	长度为2个字节。取值为0x0806	
硬件类型	长度为2个字节。表示网络类型；以太网的取值为1	

（续表）

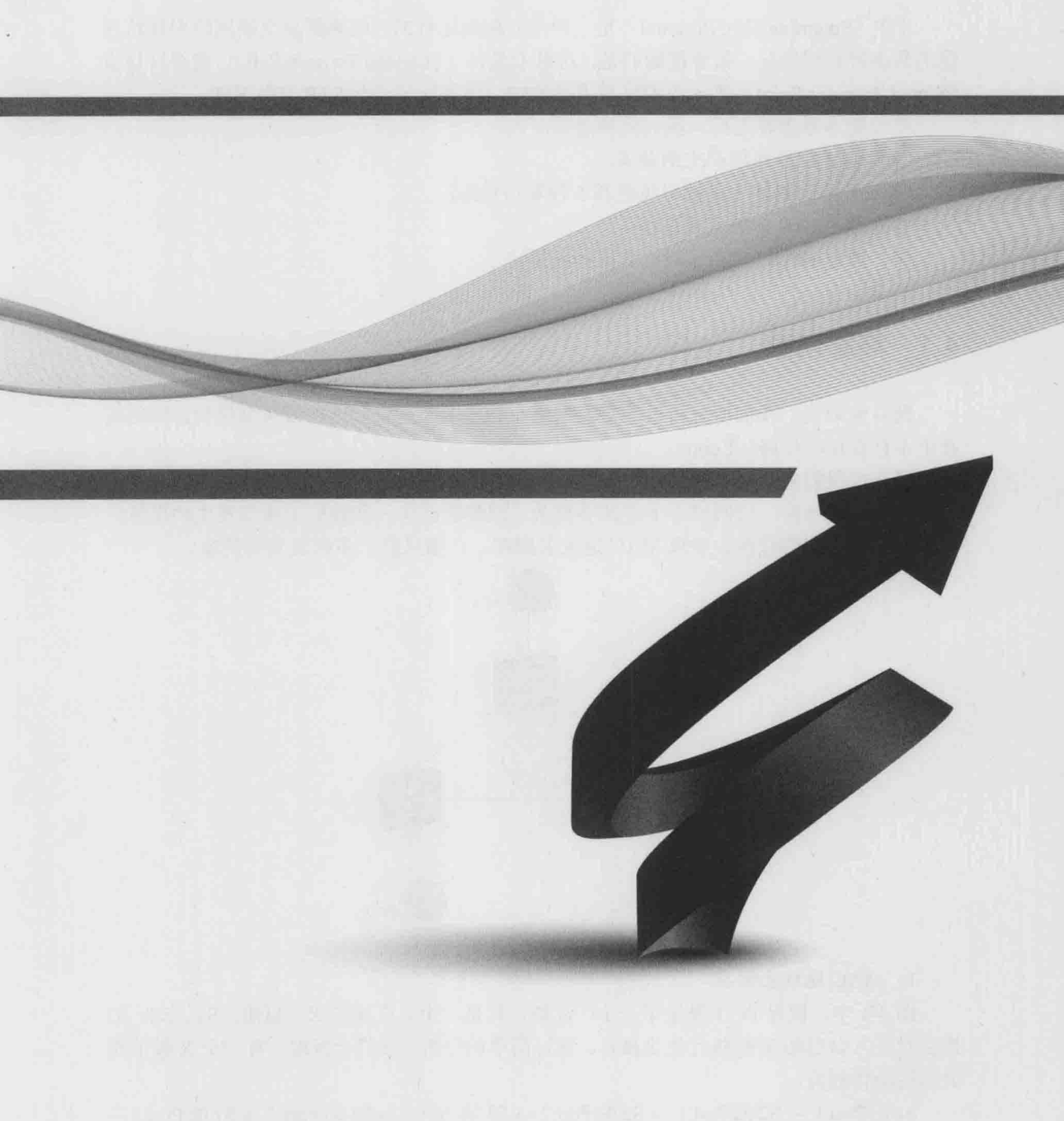
字段	ARP 请求报文	ARP 应答报文
协议类型	长度为 2 个字节。表示协议地址类型；取值为 0x0800 即表示根据 IP 地址来进行映射	
硬件地址长度（hln）	长度为 1 个字节。表示硬件地址的长度；以太网中取值为 6，表示 MAC 地址长度为 6 个字节	
协议地址长度（pln）	长度为 1 个字节。表示协议地址的长度；取值为 4 表示 IP 地址长度为 4 个字节	
op	长度为 2 个字节。表示 ARP 报文的种类；取值为 1 表示是 ARP 请求报文	长度为 2 个字节。表示 ARP 报文的种类；取值为 2 表示是 ARP 应答报文
发送端 MAC 地址	请求端的 MAC 地址	被请求端的 MAC 地址
发送端 IP 地址	请求端的 IP 地址	被请求端的 IP 地址
目的端 MAC 地址	请求端发出请求时，还不知道该 MAC 地址。接收方忽略该字段	请求端的 MAC 地址
目的端 IP 地址	请求端希望映射的 IP 地址，也就是被请求端的 IP 地址	请求端的 IP 地址
PAD	PAD 字段一共有 18 个字节，目的是为了凑足以太帧的载荷数据的最小长度 46 字节	

3.4.3 练习题

1. （多选）下列描述中正确的是？（ ）
   A．ARP 的作用是根据已知的 MAC 地址信息获取相应的 IP 地址信息
   B．ARP 的作用是根据已知的 IP 地址信息获取相应的 MAC 地址信息
   C．ARP 是一个数据链路层协议
   D．ARP 是一个网络层协议
2. （单选）携带 ARP 应答报文的帧应该是一个什么帧？（ ）
   A．广播帧　　B．组播帧　　C．单播帧
3. （单选）携带 ARP 请求报文的帧应该是一个什么帧？（ ）
   A．广播帧　　B．组播帧　　C．单播帧
4. （单选）携带 ARP 报文的帧的类型字段的值应该是多少？（ ）
   A．0x0800　　B．0x0806　　C．0x8006
5. （单选）ARP 缓存表中存放的是？（ ）
   A．IP 地址与端口编号之间的对应关系
   B．MAC 地址与 IP 地址之间的对应关系
   C．MAC 地址与端口编号之间的对应关系



第4章 STP协议



STP（Spanning Tree Protocol）是一种由交换机运行的、用来解决交换网络中环路问题的数据链路层协议。需要提醒的是，屏蔽双绞线（Shielded Twisted Pair）、信令转接点（Signal Transfer Point）等术语的缩写也是 STP，读者应分清此 STP 与彼 STP。

学习完本章内容之后，我们应该能够：

（1）清楚 STP 协议产生的背景；

（2）理解 STP 的 3 种端口角色和 5 种端口状态；

（3）熟悉 STP 树的生成过程；

（4）理解 BPDU 中各字段的含义和作用。

4.1 环路问题

到目前为止，我们所分析过的交换网络在物理上都是星型结构或树型结构，网络拓扑中不存在任何环路（Loop）。

现在，我们来看一下图 4-1 所示的网络，在这个网络中，3 台交换机 S1、S2、S3 之间形成了一个环路。该网络看上去并不复杂，但却会产生一些我们不希望发生的现象。概括地讲，环路的存在会导致 MAC 地址表翻摆、广播风暴、多帧复制等现象。

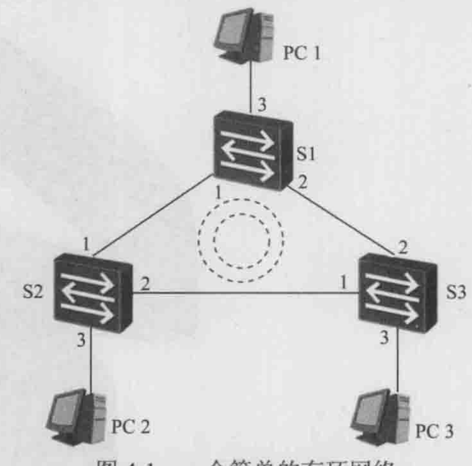


图 4-1　一个简单的有环网络

1. MAC 地址表翻摆

图 4-1 中，假设 PC 1 发送了一个（注意，只是一个）广播帧 X。显然，S1、S2、S3 都会对进入自己的 X 帧执行泛洪操作。通过简单的分析，我们会发现，有一个 X 帧的拷贝的运动轨迹为：

S1 的 Port 1 → S2 的 Port 1 → S2 的 Port 2 → S3 的 Port 1 → S3 的 Port 2 → S1 的 Port 2 → S1 的 Port 1 → S2 的 Port 1 → S2 的 Port 2 → S3 的 Port 1 → S3 的 Port 2 → S1 的 Port 2 → S1 的 Port 1……（逆时针旋转）

另外，我们还会发现，还有一个 X 帧的拷贝的运动轨迹为：

S1 的 Port 2 → S3 的 Port 2 → S3 的 Port 1 → S2 的 Port 2 → S2 的 Port 1 → S1 的 Port 1 → S1 的 Port 2 → S3 的 Port 2 → S3 的 Port 1 → S2 的 Port 2 → S2 的 Port 1 → S1 的 Port 1 → S1

的 Port 2……（顺时针旋转）

也就是说，X 的一个拷贝会永不停止地逆时针快速旋转，另一个拷贝会永不停止地顺时针快高速旋转。每当 X 的拷贝从 Port 1 进入 S1 时，S1 都会将 MAC 地址表中关于 PC 1 的 MAC 地址的表项内容修改为“PC 1 的 MAC 地址 ←→ Port 1”，而每当 X 的拷贝从 Port 2 进入 S1 时，S1 都会将 MAC 地址表中关于 PC 1 的 MAC 地址的表项内容修改为“PC 1 的 MAC 地址 ←→ Port 2”。这样一来，S1 的 MAC 地址表中关于 PC 1 的 MAC 地址的表项内容就会无休止地、快速地变来变去，这就是翻摆现象。S2、S3 的 MAC 地址表也会出现完全一样的快速翻摆现象。MAC 地址表的快速翻摆会大量消耗交换机的处理资源，甚至可能会导致交换机瘫痪。

2. 广播风暴

刚才说到，X 的一个拷贝会永不停止地逆时针快速旋转，另一个拷贝会永不停止地顺时针快高速旋转。这意味着，每台交换机都会不停地接收到 X 帧的拷贝，并且对其执行泛洪操作。显然，Si（$i=1$，2，3）每执行一次泛洪操作，PC i（$i=1$，2，3）都会收到一个 X 帧的拷贝；Si 不停地执行泛洪操作，PC i（$i=1$，2，3）不停地收到 X 帧的拷贝，这就产生了被称为广播风暴（Broadcast Storm）的现象。广播风暴会大量地消耗网络的带宽资源以及计算机的处理资源。别忘了，计算机每收到一个广播帧后，都会将该广播帧的载荷数据上送给网络层去处理。大量的广播帧来袭，很可能导致计算机瘫痪。

3. 多帧复制

图 4-1 中，假设 PC 1 向 PC 2 发送了一个单播帧 Y，并且假设 S1 的 MAC 地址表中不存在关于 PC 2 的 MAC 地址的表项，S2 的 MAC 地址表中存在表项“PC 2 的 MAC 地址←→Port 3”，S3 的 MAC 地址表中存在表项“PC 2 的 MAC 地址←→Port 1”。显然，S1 会对 Y 帧执行泛洪操作，S2 和 S3 都会对 Y 帧执行点对点转发操作。最后的结果是，PC 2 会收到两个 Y 帧的拷贝。这种现象称为多帧复制，是我们不希望发生的现象。

环路的存在，会导致 MAC 地址表翻摆、广播风暴、多帧复制等我们不希望发生的现象。那么，环路能带来一些我们希望得到的东西吗？答案是肯定的：环路能提高网络连接的可靠性。如图 4-1 所示，因为有环路的存在，即使某两台交换机之间的链路因故障而中断了，整个网络仍然会保持其连通性，而这在无环网络中是无法做到的。

为了得到环路带来的好处（提高网络连接的可靠性），同时又避免因环路而产生的灾难性问题（MAC 地址表翻摆、广播风暴、多帧复制），IEEE 802.1D 中定义了 STP（Spanning Tree Protocol）协议。在描述 STP 协议之前，我们还需要了解几个基本术语：桥、桥的 MAC 地址、桥 ID、端口 ID。

1. 桥（Bridge）

因为性能方面的限制等因素，早期的交换机一般只有两个转发端口（如果端口多了，交换的转发速度就会慢得无法接受），所以那时的交换机常常被称为“网桥”，或简称“桥”。在 IEEE 的术语中，“桥”这个术语一直沿用至今，但并不只是指只有两个转发端口的交换机了，而是泛指具有任意多端口的交换机。目前，“桥”和“交换机”这两个术语是完全混用的，本书也采用了这一混用习惯。

2. 桥的 MAC 地址（Bridge MAC Address）

我们知道，一个桥有多个转发端口，每个端口有一个 MAC 地址。通常，我们把端

口编号最小的那个端口的 MAC 地址作为整个桥的 MAC 地址。

3. 桥 ID（Bridge Identifier，BID）

如图 4-2 所示，一个桥（交换机）的桥 ID 由两部分组成，前面 2 个字节是这个桥的桥优先级，后面 6 个字节是这个桥的 MAC 地址。桥优先级的值可以人为设定，缺省值为 0x8000（相当于十进制的 32 768）。

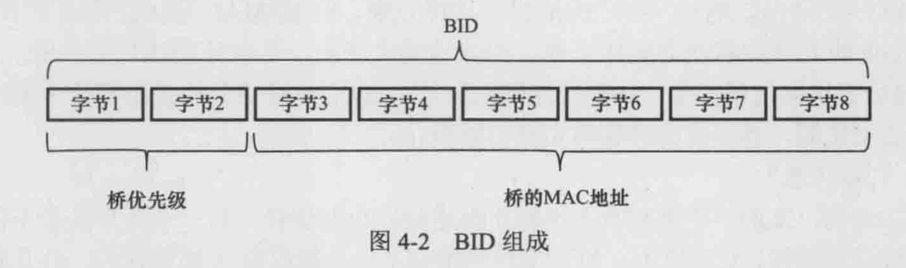


图 4-2 BID 组成

4. 端口 ID（Port Identifier，PID）

一个桥（交换机）的某个端口的端口 ID 的定义方法有很多种，图 4-3 给出了其中的两种定义。在第一种定义中，端口 ID 由两个字节组成，第一个字节是该端口的端口优先级，后一个字节是该端口的端口编号。在第二种定义中，端口 ID 由 16 个比特组成，前 4 个比特是该端口的端口优先级，后 12 比特是该端口的端口编号。端口优先级的值是可以人为设定的。不同的设备商所采用的 PID 定义方法可能不同。

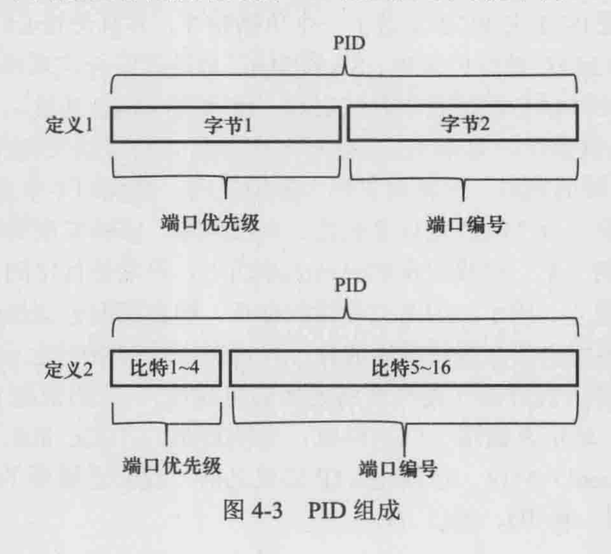


图 4-3 PID 组成

4.2 STP 树的生成

STP 协议的基本原理：在一个具有物理环路的交换网络中，交换机通过运行 STP 协议，自动生成一个没有环路的工作拓扑。该无环工作拓扑也称为 STP 树（STP Tree），树节点为某些特定的交换机，树枝为某些特定的链路。一棵 STP 树包含了唯一的一个根节点，任何一个节点到根节点的工作路径不但是唯一的，而且是最优的。当网络拓扑发生

变化时，STP 树也会自动地发生相应的改变。

简言之，有环的物理拓扑提高了网络连接的可靠性，而无环的工作拓扑避免了广播风暴、MAC 地址表翻摆、多帧复制，这就是 STP 的精髓。

STP 树的生成过程是：首先选举根桥（Root Bridge），然后确定根端口（Root Port，RP）和指定端口（Designated Port，DP），最后阻塞备用端口（Alternate Port，AP）。

4.2.1 选举根桥

根桥是 STP 树的根节点。要生成一棵 STP 树，首先要确定出一个根桥。根桥是整个交换网络的逻辑中心，但不一定是它的物理中心。当网络的拓扑发生变化时，根桥也可能会发生变化。

运行 STP 协议的交换机（简称为 STP 交换机）会相互交换 STP 协议帧，这些协议帧的载荷数据被称为 BPDU（Bridge Protocol Data Unit，网桥协议数据单元）。虽然 BPDU 是 STP 协议帧的载荷数据，但它并非网络层的数据单元；BPDU 的产生者、接收者、处理者都是 STP 交换机本身，而非终端计算机。BPDU 中包含了与 STP 协议相关的所有信息（后续会对 BPDU 进行专门的讲解），其中就有 BID。

STP 交换机初始启动之后，都会认为自己是根桥，并在发送给别的交换机的 BPDU 中宣告自己是根桥。当交换机从网络中收到其他设备发送过来的 BPDU 的时候，会比较 BPDU 中指定的根桥 BID 和自己的 BID。交换机不断地交互 BPDU，同时对 BID 进行比较，直至最终选举出一台 BID 最小的交换机作为根桥。

如图 4-4 所示，交换机 S1、S2、S3 都使用了默认的桥优先级 32768。显然，S1 的 BID 最小，所以最终 S1 将被选举为根桥。

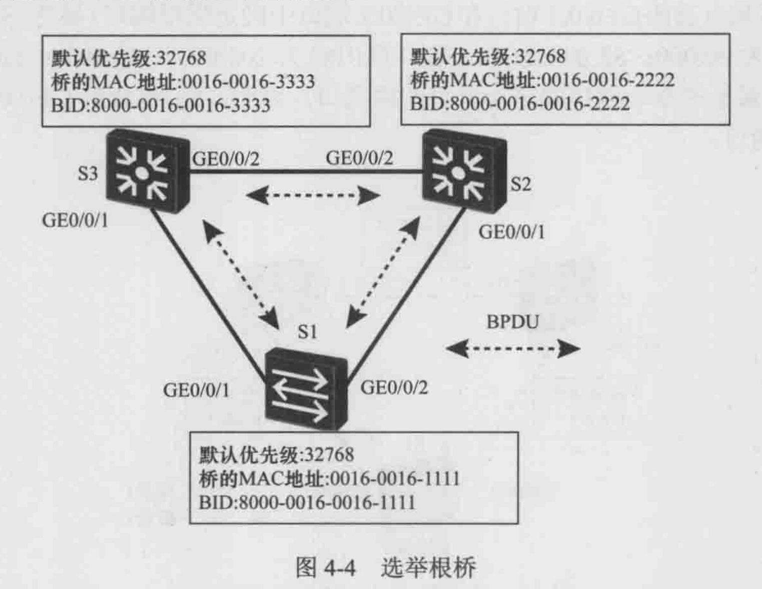


图 4-4 选举根桥

4.2.2 确定根端口

根桥确定后，其他没有成为根桥的交换机都被称为非根桥。一台非根桥设备上可能

会有多个端口与网络相连，为了保证从某台非根桥设备到根桥设备的工作路径是最优且唯一的，就必须从该非根桥设备的端口中确定出一个被称为“根端口”的端口，由根端口来作为该非根桥设备与根桥设备之间进行报文交互的端口。一台非根桥设备上最多只能有一个根端口。

STP 协议把根路径开销作为确定根端口的一个重要依据。一个运行 STP 协议的网络中，我们将某个交换机的端口到根桥的累计路径开销（即从该端口到根桥所经过的所有链路的路径开销的和）称为这个端口的根路径开销（Root Path Cost，RPC）。链路的路径开销（Path Cost）与端口速率有关，端口转发速率越大，则路径开销越小。端口速率与路径开销的对应关系可参考表 4-1。

表 4-1　　端口速率与路径开销的对应关系

端口速率	路径开销（IEEE 802.1t 标准）
10Mbit/s	2 000 000
100Mbit/s	200 000
1Gbit/s	20 000
10Gbit/s	2 000

说明：

此表的内容是 IEEE 802.1t 标准定义的。在实际中，不同的设备商所采用的标准可能不同。

如图 4-5 所示，假定 S1 已被选举为根桥，并且链路的路径开销遵从 IEEE 802.1t，现在，S3 需要从自己的 GE0/0/1 端口和 GE0/0/2 端口中确定出根端口。显然，S3 的 GE0/0/1 端口的 RPC 为 20 000；S3 的 GE0/0/2 端口的 RPC 为 200 000 + 20 000 = 220 000。交换机会将 RPC 最小的那个端口确定为自己的根端口。因此，S3 将会把 GE0/0/1 端口确定为自己的根端口。

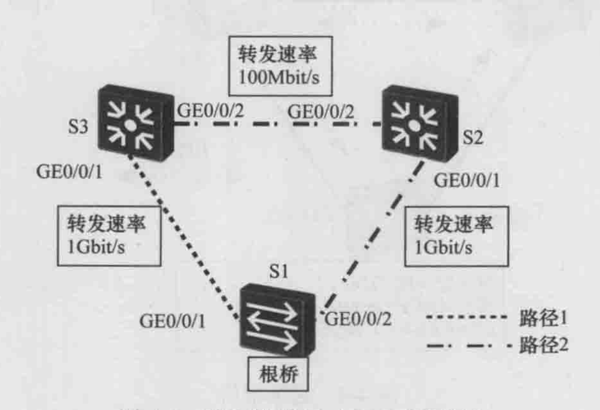


图 4-5　确定根端口（RPC 不同时）

然而，一台非根桥设备上不同端口的 RPC 可能相同。在这种情况下，就必须按照图 4-6 所示的流程来确定根端口。

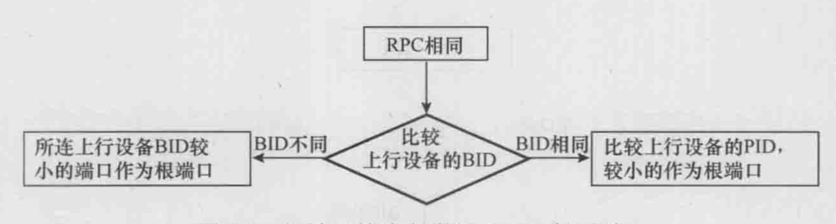


图 4-6　根端口的确定流程（RPC 相同时）

如图 4-7 所示，S1 是根桥，假设 S4 的 GE0/0/1 端口的 RPC（路径 1 的开销）与 GE0/0/2 端口的 RPC（路径 2 的开销）相同，则 S4 会对上行设备 S2 和 S3 的 BID 进行比较：如果 S2 的 BID 小于 S3 的 BID，则 S4 会将自己的 GE0/0/1 端口确定为自己的根端口；如果 S3 的 BID 小于 S2 的 BID，则 S4 会将自己的 GE0/0/2 端口确定为自己的根端口。对于 S5 而言，假设其 GE0/0/1 端口的 RPC 与 GE0/0/2 端口的 RPC 相同，由于这两个端口的上行设备同为 S4，所以 S5 还会对 S4 的 GE0/0/3 端口的 PID 和 S4 的 GE0/0/4 端口的 PID 进行比较：如果 S4 的 GE0/0/3 端口的 PID 小于 S4 的 GE0/0/4 端口的 PID，则 S5 会将自己的 GE0/0/1 端口确定为自己的根端口；如果 S4 的 GE0/0/4 端口的 PID 小于 S4 的 GE0/0/3 端口的 PID，则 S5 会将自己的 GE0/0/2 端口确定为自己的根端口。

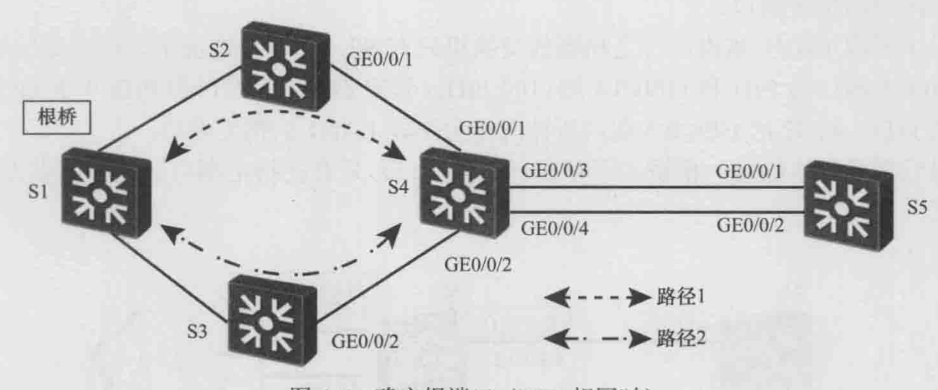


图 4-7　确定根端口（RPC 相同时）

4.2.3　确定指定端口

根端口保证了交换机与根桥之间工作路径的唯一性和最优性。为了防止工作环路的存在，网络中每个网段与根桥之间的工作路径也必须是唯一的且最优的。当一个网段有两条及两条以上的路径通往根桥时（该网段连接了不同的交换机，或者该网段连接了同一台交换机的不同端口），与该网段相连的交换机（可能不止一台）就必须确定出一个唯一的指定端口。

指定端口也是通过比较 RPC 来确定的，RPC 较小的端口将成为指定端口。如果 RPC 相同，则需要比较 BID、PID 等，具体流程如图 4-8 所示。

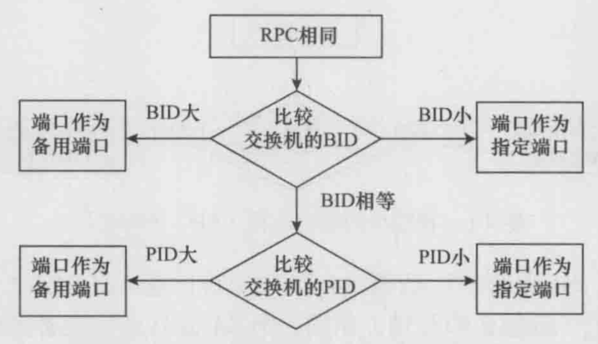


图 4-8　指定端口的确定流程（RPC 相同时）

如图 4-9 所示，假定 S1 已被选举为根桥，并且假定各链路的开销均相等。显然，S3 的 GE0/0/1 端口的 RPC 小于 S3 的 GE0/0/2 端口的 RPC，所以 S3 将自己的 GE0/0/1 端口确定为自己的根端口。类似地，S2 的 GE0/0/1 端口的 RPC 小于 S2 的 GE0/0/2 端口的 RPC，所以 S2 将自己的 GE0/0/1 端口确定为自己的根端口。

对于 S3 的 GE0/0/2 和 S2 的 GE0/0/2 之间的网段来说，S3 的 GE0/0/2 端口的 RPC 是与 S2 的 GE0/0/2 端口的 RPC 相等的，所以需要比较 S3 的 BID 和 S2 的 BID。假定 S2 的 BID 小于 S3 的 BID，则 S2 的 GE0/0/2 端口将被确定为 S3 的 GE0/0/2 和 S2 的 GE0/0/2 之间的网段的指定端口。

对于网段 LAN1 来说，与之相连的交换机只有 S2。在这种情况下，就需要比较 S2 的 GE0/0/3 端口的 PID 和 GE0/0/4 端口的 PID。假定 GE0/0/3 端口的 PID 小于 GE0/0/4 端口的 PID，则 S2 的 GE0/0/3 端口将被确定为网段 LAN1 的指定端口。

最后需要指出的是，根桥上不存在任何根端口，只存在指定端口。读者可以去想想为什么。

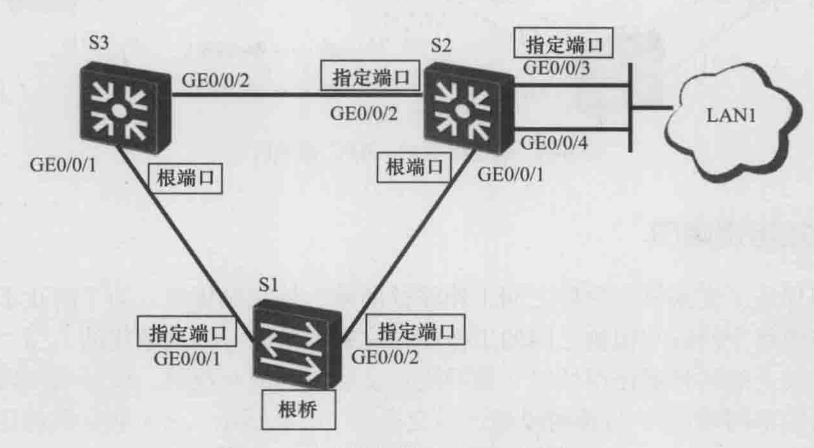


图 4-9　确定指定端口（RPC 相同时）

4.2.4　阻塞备用端口

在确定了根端口和指定端口之后，交换机上所有剩余的非根端口和非指定端口统称

为备用端口。STP 会对这些备用端口进行逻辑阻塞。所谓逻辑阻塞，是指这些备用端口不能转发由终端计算机产生并发送的帧，这些帧也被称为用户数据帧。不过，备用端口可以接收并处理 STP 协议帧。根端口和指定端口既可以发送和接收 STP 协议帧，又可以转发用户数据帧。

如图 4-10 所示，一旦备用端口被逻辑阻塞后，STP 树（无环工作拓扑）的生成过程便告完成。

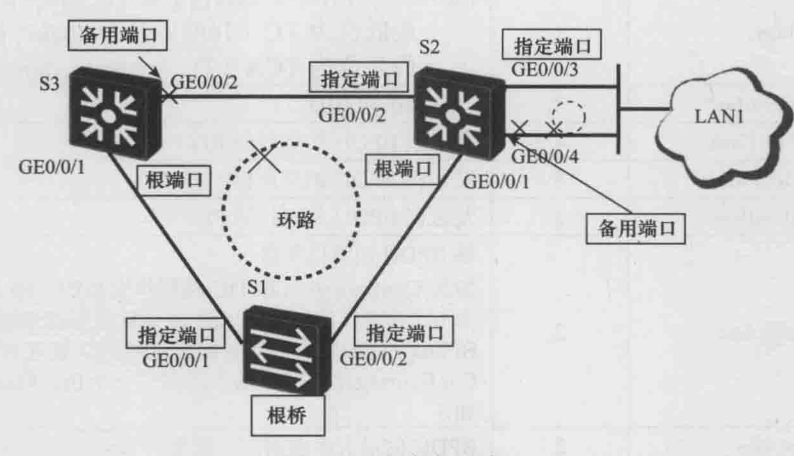


图 4-10　阻塞备用端口

4.3　STP 报文格式

STP 交换机通过交换 STP 协议帧来建立和维护 STP 树，并在网络的物理拓扑发生变化时重建新的 STP 树。

STP 协议帧由 STP 交换机产生、发送、接收、处理。STP 协议帧是一种组播帧，组播地址为 01-80-c2-00-00-00。

STP 协议帧采用了 IEEE 802.3 封装格式，其载荷数据被称为 BPDU。BPDU 有两种类型：Configuration BPDU 和 TCN（Topology Change Notification）BPDU。

4.3.1　Configuration BPDU

在初始形成 STP 树的过程中，各 STP 交换机都会周期性地（缺省为 2s）主动产生并发送 Configuration BPDU。在 STP 树形成后的稳定期，只有根桥才会周期性地（缺省为 2s）主动产生并发送 Configuration BPDU；相应地，非根交换机会从自己的根端口周期性地接收到 Configuration BPDU，并立即被触发而产生自己的 Configuration BPDU，且从自己的指定端口发送出去。这一过程看起来就像是根桥发出的 Configuration BPDU 逐跳地“经过”了其他的交换机。

Configuration BPDU 的格式如表 4-2 所示。

表 4-2 **BPDU 的格式**

字段	字节数	简单说明
Protocol Identifier	2	总是为 0x0000
Protocol Version Identifier	1	总是为 0x00
BPDU Type	1	BPDU 类型： 0x00: Configuration BPDU； 0x80: TCN BPDU
Flags	1	网络拓扑变化标志：仅使用了最低位和最高位 最低位为 TC（Topology Change）标志； 最高位为 TCA（TC Acknowledgment）标志
Root Identifier	8	当前根桥的 BID
Root Path Cost	4	发送该 BPDU 的端口的 RPC
Bridge Identifier	8	发送该 BPDU 的交换机的 BID
Port Identifier	2	发送该 BPDU 的端口的 PID
Message Age	2	该 BPDU 消息的年龄 如果 Configuration BPDU 是根桥发出的，则 Message Age 为 0。否则，Message Age 是从根桥发送到当前桥接收到 BPDU 的总时间，包括传输延时等。在实际的实现中，Configuration BPDU 每“经过”一个桥，Message Age 增加 1
Max Age	2	BPDU 的最大生命周期，缺省为 20s
Hello Time	2	根桥发送 Configuration BPDU 的周期，也相应地成为了其他交换机发送 Configuration BPDU 的周期，缺省为 2s
Forward Delay	2	控制端口 Listening 和 Learning 状态的持续时间，缺省为 15s

Configuration BPDU 中携带的参数可以分为 3 类：第一类是 BPDU 对自身的标识，包括协议标识、版本号、BPDU 类型和 Flags；第二类是用于进行 STP 计算的参数，包括发送该 BPDU 的交换机的 BID，当前根桥的 BID，发送该 BPDU 的端口的 PID，以及发送该 BPDU 的端口的 RPC；第三类是时间参数，分别是 Hello Time、Forward Delay、Message Age、Max Age。

Hello Time：交换机发送 Configuration BPDU 的时间间隔。当网络拓扑及 STP 树稳定之后，全网使用根桥指定的 Hello Time。如果要修改该时间参数，则必须在根桥上修改才有效。

Forward Delay：端口状态迁移的延迟时间。STP 树的生成需要一定的时间，在此过程中各交换机的端口状态的变化并不是同步的。如果新选出的根端口和指定端口立刻就开始进行用户数据帧的转发的话，可能会造成临时工作环路。为此，STP 引入了 Forward Delay 机制：新选出的根端口和指定端口需要经过 2 倍的 Forward Delay 延时后才能进入用户数据帧的转发状态，以保证此时的工作拓扑已无环路。

Message Age：是指从根桥发出某个 Configuration BPDU，一直到这个 Configuration BPDU “传”到当前交换机时所需要的总的时间，包括传输延时等。实际的实现中，Configuration BPDU 每“经过”一个桥，Message Age 增加 1。从根桥发出的 Configuration BPDU 的 Message Age 为 0。

Max Age：Configuration BPDU 的最大生命周期。Max Age 的值由根桥指定，缺省值

为 20s。STP 交换机在收到 Configuration BPDU 后，会对其中的 Message Age 和 Max Age 进行比较。如果 Message Age 小于等于 Max Age，则该 Configuration BPDU 会触发该交换机产生并发送新的 Configuration BPDU，否则该 Configuration BPDU 会被丢弃（忽略），并且不会触发该交换机产生并发送新的 Configuration BPDU。

4.3.2 TCN BPDU

TCN BPDU 的结构和内容非常简单，它只有表 4-2 中列出的前 3 个字段：协议标识、版本号和类型，其中类型字段的值是 0x80。

如果网络中某条链路发生了故障，导致工作拓扑发生了改变，则位于故障点的交换机可以通过端口状态直接感知到这种变化，但是其他的交换机是无法直接感知到这种变化的。这时，位于故障点的交换机会以 Hello Time 为周期通过其根端口不断向上游交换机发送 TCN BPDU，直到接收到从上游交换机发来的、TCA 标志置 1 的 Configuration BPDU。上游交换机在收到 TCN BPDU 后，一方面会通过其指定端口回复 TCA 标志置 1 的 Configuration BPDU，另一方面会以 Hello Time 为周期通过其根端口不断向它的上游交换机发送 TCN BPDU。此过程一直重复，直到根桥接收到 TCN BPDU。根桥接收到 TCN BPDU 后，会发送 TC 标志置 1 的 Configuration BPDU，通告所有交换机网络拓扑发生了变化。图 4-11 示意了这一过程。

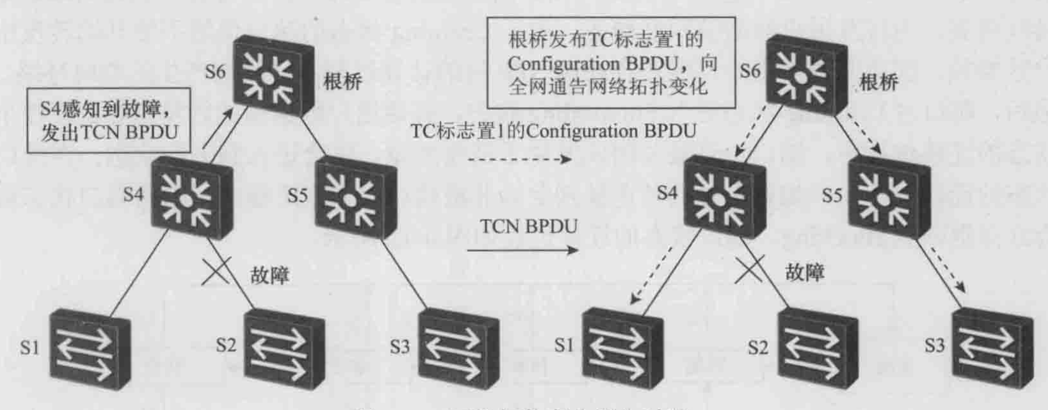


图 4-11 网络拓扑变化通告过程

交换机收到 TC 标志置 1 的 Configuration BPDU 后，便意识到网络拓扑已经发生了变化，这说明自己的 MAC 地址表的表项内容很可能已经不再是正确的了，这时交换机会将自己的 MAC 地址表的老化周期（缺省为 300s）缩短为 Forward Delay 的时间长度（缺省为 15s），以加速老化掉原来的地址表项。

4.4 STP 端口状态

通过前面的学习我们知道，STP 定义了 3 种端口角色：根端口、指定端口、备用端口。不仅如此，根据端口是否能接收和发送 STP 协议帧，以及端口是否能转发用户数据帧，STP 还将端口的状态分为了 5 种：去能状态、阻塞状态、侦听状态、学习状态、转

发状态。表 4-3 给出了这 5 种端口状态的简单说明。

表 4-3　　STP 端口的 5 种状态

端口状态	说明
去能（Disabled）	去能状态的端口无法接收和发出任何帧，端口处于关闭（Down）状态
阻塞（Blocking）	阻塞状态的端口只能接收 STP 协议帧，不能发送 STP 协议帧，也不能转发用户数据帧
侦听（Listening）	侦听状态的端口可以接收并发送 STP 协议帧，但不能进行 MAC 地址学习，也不能转发用户数据帧
学习（Learning）	学习状态的端口可以接收并发送 STP 协议帧，也可以进行 MAC 地址学习，但不能转发用户数据帧
转发（Forwarding）	转发状态的端口可以接收并发送 STP 协议帧，也可以进行 MAC 地址学习，同时能够转发用户数据帧

STP 交换机的端口在初始启动时，首先会从 Disabled 状态进入到 Blocking 状态。在 Blocking 状态，端口只能接收和分析 BPDU，但不能发送 BPDU。如果端口被选为根端口或指定端口，则会进入 Listening 状态，此时端口接收并发送 BPDU，这种状态会持续一个 Forward Delay 的时间长度，缺省为 15s。然后，如果没有因“意外情况”而回到 Blocking 状态，则该端口会进入到 Learning 状态，并在此状态持续一个 Forward Delay 的时间长度。处于 Learning 状态的端口可以接收和发送 BPDU，同时开始构建 MAC 地址映射表，为转发用户数据帧做好准备。处于 Learning 状态的端口仍然不能开始转发用户数据帧，因为此时网络中可能还存在因 STP 树的计算过程不同步而产生的临时环路。最后，端口由 Learning 状态进入 Forwarding 状态，开始用户数据帧的转发工作。在整个状态的迁移过程中，端口一旦被关闭或发生了链路故障，就会进入到去能状态；在端口状态的迁移过程中，如果端口的角色被判定为非根端口或非指定端口，则其端口状态就会立即退回到 Blocking。端口状态的迁移过程如图 4-12 所示。

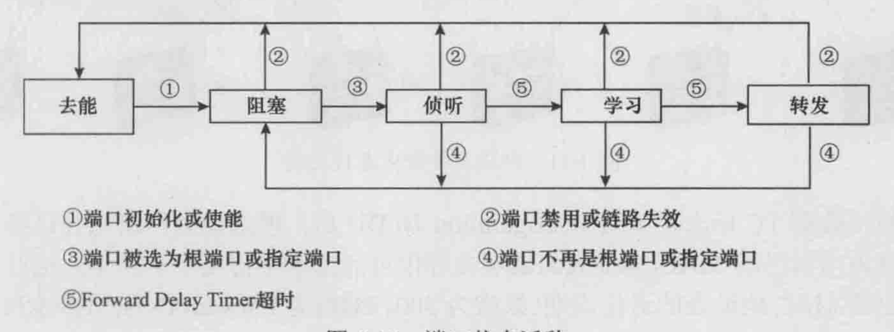


图 4-12　端口状态迁移

说明：

华为交换机在实现 STP 时，端口的状态采用了 MSTP 定义的 3 种状态：Discarding、Learning、Forwarding。

下面，我们用一个简单的例子来粗略地说明一下端口状态是如何迁移的，如图 4-13 所示。

（1）假设交换机 S1、S2、S3 大概在同一时刻启动，各交换机的各个端口立即从 Disabled 状态进入到 Blocking 状态。由于处于 Blocking 状态的端口只能接收而不能发送 BPDU，所以任何端口都收不到 BPDU。在等待 Max Age 的时间（缺省为 20s）后，每台交换机都会认为自己就是根桥，所有端口的角色都会成为指定端口，并且端口的状态迁移为 Listening。

（2）交换机的端口进入 Listening 状态后，开始发送自己产生的 Configuration BPDU，同时也接收其他交换机发送的 Configuration BPDU。

假定 S2 最先发送 Configuration BPDU，当 S3 从自己的 GE0/0/2 端口收到 S2 发送的 Configuration BPDU 后，会认为 S2 才应该是根桥（因为 S2 的 BID 小于 S3 的 BID），于是 S3 会把自己的 GE0/0/2 端口由指定端口变更为根端口，然后将自己重新产生的、根桥设置为 S2 的 Configuration BPDU 从自己的 GE0/0/1 端口发送出去。

当 S1 从自己的 GE0/0/1 端口接收到 S3 发送过来的 Configuration BPDU 后，会发现自己的 BID 才是最小的，自己更应该成为根桥，于是立即向 S3 发去自己的 Configuration BPDU。当然，如果 S1 从自己的 GE0/0/2 端口接收到 S2 发送过来的 Configuration BPDU，也会立即向 S2 发去自己的 Configuration BPDU。

S2 和 S3 收到 S1 发送的 Configuration BPDU 后，会确认 S1 就是根桥，于是 S2 的 GE0/0/1 端口和 S3 的 GE0/0/1 端口都会成为根端口，S2 和 S3 会从各自的 GE0/0/2 端口发送新的 Configuration BPDU。然后，S3 的 GE0/0/2 端口会成为备用端口，进入 Blocking 状态，S2 的 GE0/0/2 端口仍然为指定端口。

因为各交换机发送 BPDU 的时间先后带有一定的随机性，所以上述的过程并不是唯一的。但是，无论各个交换机端口最开始的状态如何，也无论中间的过程差异如何，最终的结果总是确定而唯一的：BID 最小的交换机会成为根桥，各端口的角色会变化成为自己应该扮演的角色。

端口在 Listening 状态持续 Forward Delay 的时间长度（缺省 15s）后，开始进入 Learning 状态。注意，S3 的 GE0/0/2 端口已经变成了备用端口，所以其状态会成为 Blocking 状态。

（3）各个端口（S3 的 GE0/0/2 端口除外）相继进入 Learning 状态后，会持续 Forward Delay 的时间长度（缺省 15s）。在此时间内，交换机可以开始学习 MAC 地址与这些端口的映射关系，同时希望 STP 树在这段时间内能够完全收敛。

（4）然后，各端口（S3 的 GE0/0/2 端口除外）相继进入 Forwarding 状态，开始用户数据帧的转发工作。

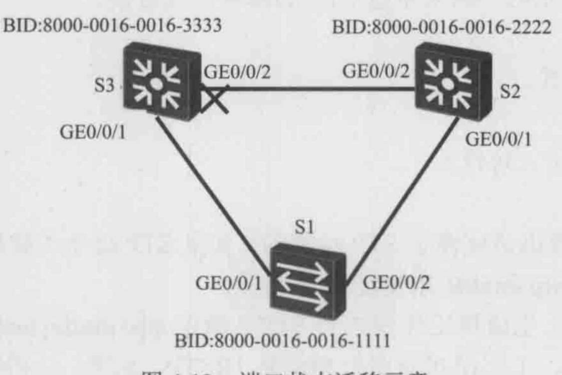


图 4-13　端口状态迁移示意

4.5 STP 的改进

STP 网络中，STP 树的完全收敛需要依赖定时器的计时，端口状态从 Blocking 迁移到 Forwarding 至少需要两倍 Forward Delay 的时间长度，总的收敛时间太长，一般需要几十秒的时间。为了弥补 STP 慢收敛的缺陷，IEEE 802.1w 定义了 RSTP（Rapid Spanning Tree Protocol）。RSPT 在 STP 的基础上进行了许多改进，使得收敛时间大大减少，一般只需要几秒钟的时间。在现实网络中，STP 几乎已经停止使用，取而代之的是 RSTP。

下面我们简单介绍一下 RSTP 改进点中的两个。

1. 3 种端口状态

RSTP 中，端口的状态只有 3 种：Discarding、Learning、Forwarding。对这 3 种端口状态的说明如表 4-4 所示。

表 4-4 **RSTP 与 STP 端口状态对比**

RSTP 端口状态	对应的 STP 端口状态	说明
Forwarding	Forwarding	可以转发用户数据帧；可以学习 MAC 地址
Learning	Learning	不能转发用户数据帧，但是可以学习 MAC 地址
Discarding	Listening	不能转发用户数据帧，也不能学习 MAC 地址
Discarding	Blocking	
Discarding	Disabled	

2. P/A 机制

STP 计算中，一个端口在成为指定端口后，需要等待至少两倍 Forward Delay 的时间才可能进入 Forwarding 状态。而在 RSTP 计算中，一个端口成为指定端口之后，此端口会先进入到 Discarding 状态，然后采用 Proposal/Agreement 机制（简称 P/A 机制）主动与对端端口进行协商，通过协商并进行相关动作后，就可以立即进入 Forwarding 状态。

关于 RSTP 的深入学习和讨论，以及关于 MSTP（Multiple Spanning Tree Protocol）内容的学习，已经超出了本书的知识范围，这里略去不讲。

4.6 STP 配置示例

我们以图 4-14 所示的网络来示意 STP 的基本配置方法。

1. 配置思路

（1）配置 STP 模式。

（2）指定根桥。

（3）指定备份根桥（可选）。

2. 配置步骤

默认情况下，交换机是使能了 STP 功能的。如果 STP 处于去使能状态，需要首先在系统视图下使用命令 **stp enable** 来使能 STP 功能。

#配置交换机 S1 上生成树工作模式为 STP。命令 **stp mode**{**mstp**|**rstp**|**stp**}用来配置设备 STP 的工作模式。工作模式分别为 MSTP、RSTP、STP；缺省模式为 MSTP。

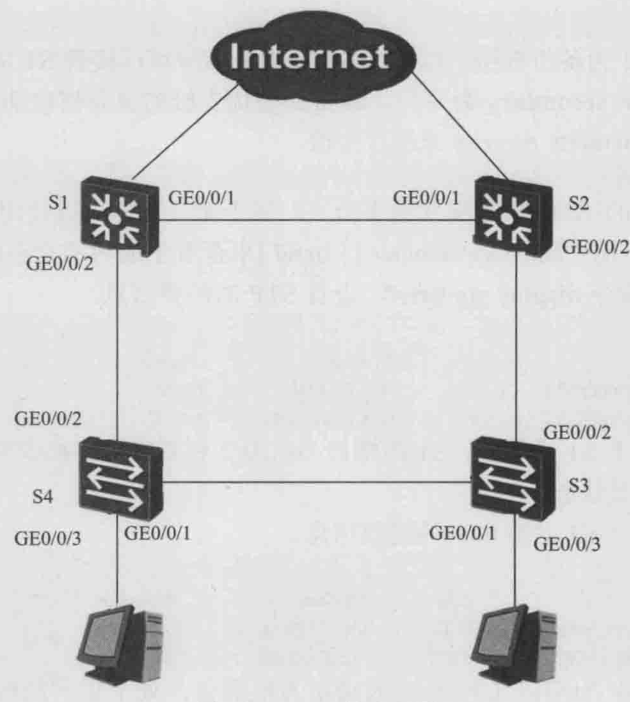


图 4-14　STP 基本配置

```
<Quidway> system-view
[Quidway] sysname S1
[S1] stp mode stp
```

#配置 S2 上生成树工作模式为 STP。

```
<Quidway> system-view
[Quidway] sysname S2
[S2] stp mode stp
```

#配置 S3 上生成树工作模式为 STP。

```
<Quidway> system-view
[Quidway] sysname S3
[S3] stp mode stp
```

#配置 S4 上生成树工作模式为 STP。

```
<Quidway> system-view
[Quidway] sysname S4
[S4] stp mode stp
```

虽然 STP 会自动选举出根桥，但通常情况下，我们会事先指定性能较好、距离网络中心较近的交换机作为根桥。本例中的网络结构非常简单，S1 和 S2 均与 Internet 相连，并且是核心交换机，S3 和 S4 是接入交换机。我们可以通过修改 S1 的桥优先级来保证 S1 被选举成为根桥。命令 **stp priority** *priority* 用来设置设备的桥优先级，*priority* 的取值范围是 0～61 440，步长为 4 096，如 0、4 096、8 192 等；缺省值是 32 768。*priority* 越小，设备被选举为根桥的可能性越大。另外，还有一种便捷的方法来指定 S1 为根桥，即通过命令 **stp root primary** 直接指定 S1 为根桥。设备上配置了此命令后，设备的桥优先级的值会被自动设为 0，并且不能通过命令 **stp priority** *priority* 来更改该设备的桥优先级。

```
[S1] stp root primary
```

接下来指定 S2 为备份根桥，以便当 S1 发生故障时可以接替 S1 成为新的根桥。在设备上执行 **stp root secondary** 命令后，设备的桥优先级的值会被自动设为 4 096，并且不能通过命令 **stp priority** *priority* 来进行修改。

```
[S2] stp root secondary
```

至此，该网络的 STP 基本配置便告结束。接下来，我们可以使用命令 **display stp** [**interface** *interface-type interface-number*] [**brief**]来查看生成树的状态信息与统计信息。

在 S1 上使用命令 **display stp brief**，查看 STP 的简要信息。

```
[S1] display stp brief
MSTID  Port                    Role   STP State      Protection
    0  GigabitEthernet0/0/1    DESI   FORWARDING     NONE
    0  GigabitEthernet0/0/2    DESI   FORWARDING     NONE
```

可以看到，由于 S1 是根桥，S1 的端口 GE0/0/2 和 GE0/0/1 都成为了指定端口，并且均处于正常的转发状态。

我们再查看一下 S4 上的 STP 的简要信息。

```
[S4] display stp brief
MSTID  Port                    Role   STP State      Protection
    0  GigabitEthernet0/0/1    ALTE   DISCARDING     NONE
    0  GigabitEthernet0/0/2    ROOT   FORWARDING     NONE
```

可以看到， S4 的端口 GE0/0/2 被确定为根端口，处于正常的转发状态，但它的 GE0/0/1 端口被阻塞，成为了备用端口。

4.7 练习题

1. （单选）关于 STP，下列描述正确的是？（ ）
   A. STP 是数据链路层协议
   B. STP 是网络层协议
   C. STP 是传输层协议
2. （单选）关于 STP，下列描述正确的是？（ ）
   A. STP 树的收敛过程通常需要几十分钟
   B. STP 树的收敛过程通常需要几十秒钟
   C. STP 树的收敛过程通常需要几秒钟
3. （多选）关于 STP，下列描述正确的是？（ ）
   A. 根桥上不存在指定端口
   B. 根桥上不存在根端口
   C. 一个非根桥上可能存在一个根端口和多个指定端口
   D. 一个非根桥上可能存在多个根端口和一个指定端口
4. （多选）关于 STP，下列描述正确的是？（ ）
   A. 端口的状态有可能从 Listening 状态直接迁移到 Forwarding 状态
   B. 端口的状态有可能从 Learning 状态直接退回到 Listening 状态
   C. 处于 Blocking 状态的端口是不能接收和发送 STP 协议帧的

D．处于 Blocking 状态的端口是不能转发用户数据帧的

5．（多选）关于 STP，下列描述正确的是？（　）

A．根桥不可能发送 TCN BPDU

B．非根桥不可能发送 TC 标志置 1 的 Configuration BPDU

C．STP 协议帧是单播帧

D．STP 协议帧是组播帧

6．（多选）关于 STP，下列描述正确的是？（　）

A．桥优先级的值越小，则桥优先级越高

B．一个交换机的 BID（Bridge Identifier）的值越小，则它成为根桥的可能性就越大

C．PID 的值不会影响根桥的选举结果

D．非根桥上可能既无根端口，也无指定端口

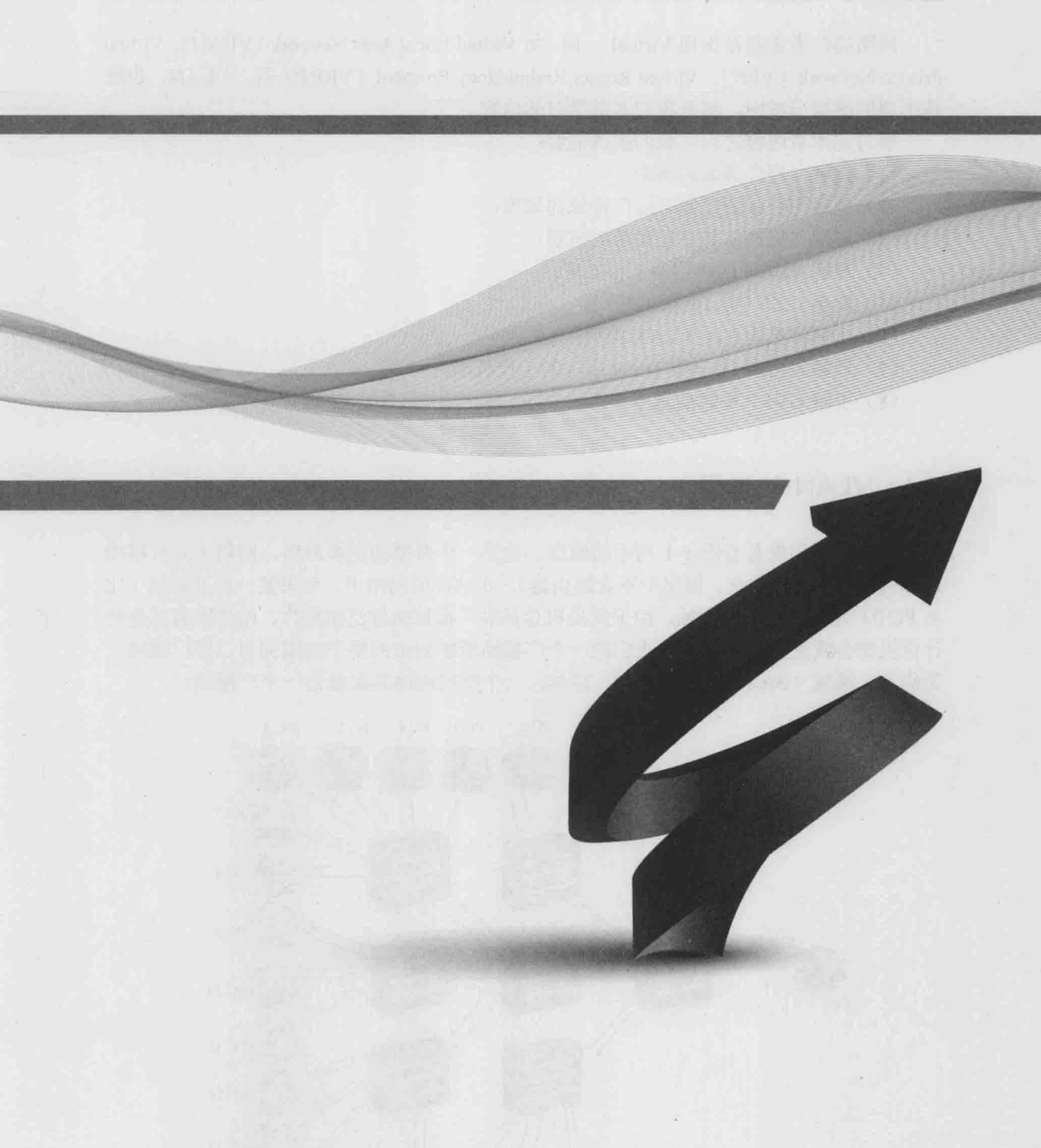
7．（多选）关于 STP，下列描述正确的是？（　）

A．Hello Time 的缺省时长是 2s

B．Max Age 的缺省时长是 15s

C．Forward Delay 的缺省时长是 20s

第5章
VLAN



网络通信术语常常出现 Virtual 一词，如 Virtual Local Area Network（VLAN）、Virtual Private Network（VPN）、Virtual Router Redundancy Protocol（VRRP）等。VLAN，也就是所谓的虚拟局域网，将是我们本章学习的内容。

学习完本章内容之后，我们应该能够：

（1）理解二层广播域的概念；

（2）理解为什么要控制二层广播域的规模；

（3）理解二层通信与三层通信的区别；

（4）理解 VLAN 的作用和工作原理；

（5）熟悉 IEEE 802.1Q 帧的格式；

（6）了解常见的几种 VLAN 类型；

（7）理解 Access 端口和 Trunk 端口的工作机制；

（8）了解 GVRP 协议的基本作用。

5.1 VLAN 的作用

首先，我们来看看图 5-1 所示的网络。这是一个典型的交换网络，网络中只有终端计算机和交换机（注意，网络中不含路由器）。在这样的网络中，如果某一台计算机（比如 PC 0）发送了一个广播帧，由于交换机总是对广播帧执行泛洪操作，结果所有其他的计算机都会收到这个广播帧。我们把一个广播帧所能到达的整个范围称为二层广播域，简称为广播域（Broadcast Domain）。显然，一个交换网络其实就是一个广播域。

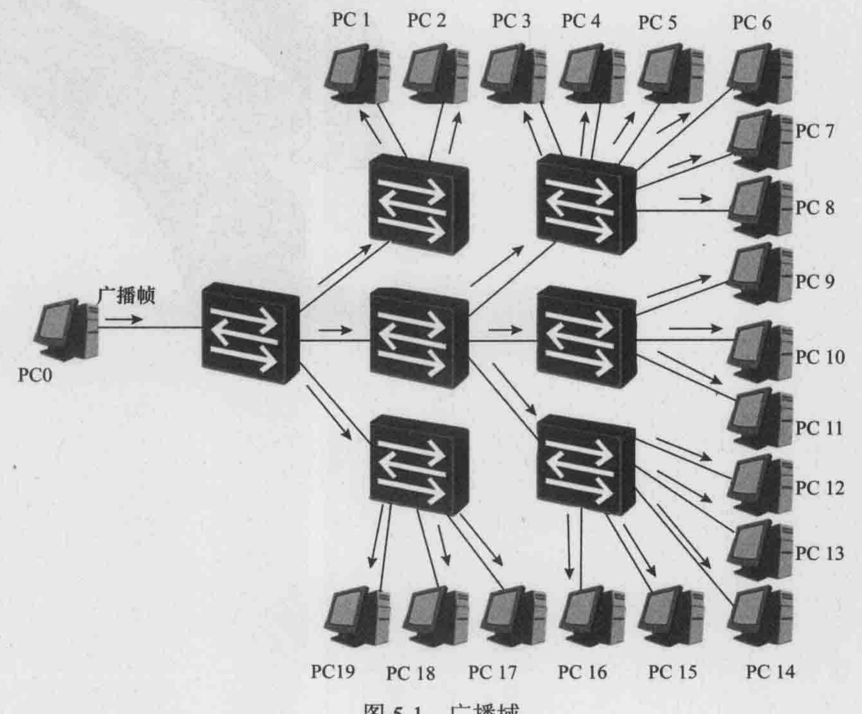


图 5-1　广播域

图 5-2 所示的网络与图 5-1 所示的网络是同一个网络。在图 5-2 中，我们假定 PC 0 向 PC 10 发送了一个单播帧 Y。假定此时 S1、S3、S7 的 MAC 地址表中存在关于 PC 10 的 MAC 地址的表项，但 S2 和 S5 的 MAC 地址表中不存在关于 PC 10 的 MAC 地址的表项，那么，S1 和 S3 将对 Y 帧执行点到点转发操作，S7 将对 Y 帧执行丢弃操作，S2 和 S5 将对 Y 帧执行泛洪操作。最后的结果是，虽然目的主机 PC 10 接收到了它应该接收到的 Y 帧，但同时 PC 3、PC 4、PC 5、PC 6、PC 7、PC 8 这几个非目的主机也接收到了它们不应该接收到的 Y 帧。这个例子向我们展示了两个我们不希望发生的问题，即一个是网络安全问题，一个是垃圾流量的问题。如果计算机可以轻易地接收到不应该接收的帧，那么就会存在安全隐患。在这个例子中，假设 PC 8 是一台恶意计算机，那么 PC 8 就轻易地窃取到了 PC 0 发送给 PC 10 的信息。另一个问题是垃圾流量问题，垃圾流量会浪费网络的带宽资源以及计算机的处理资源。在图 5-2 中，垃圾流量以虚箭头表示。

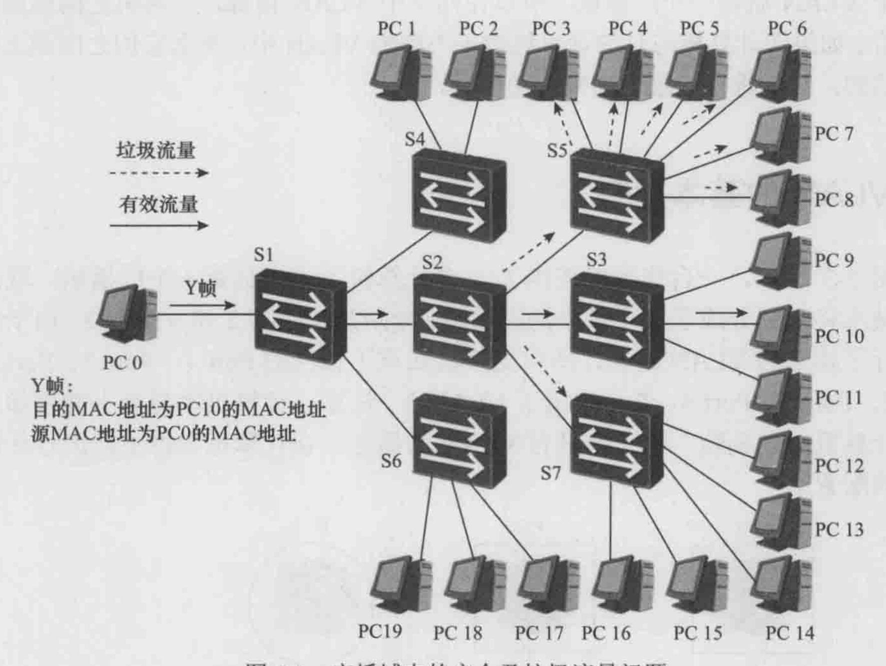


图 5-2　广播域中的安全及垃圾流量问题

显然，广播域越大，上述安全问题和垃圾流量问题就会越严重。为此，人们引入了 VLAN 技术：通过在交换机上部署 VLAN 机制，可以将一个规模较大的广播域在逻辑上划分成若干个不同的、规模较小的广播域，由此便可以有效地提升网络的安全性，同时减少了垃圾流量，节约了网络资源。

VLAN（Virtual Local Area Network，Virtual LAN）一词中，LAN 是一个转意词，用来专门指代一个广播域，而不再强调它是一个地理覆盖范围较小的局域网。谈论 VLAN 技术时，我们通常把划分前的、规模较大的广播域称为 LAN，而把划分后、规模较小的每一个广播域称为一个 Virtual LAN 或 VLAN。例如，当我们说把一个规模较大的广播域划分成了 4 个规模较小的广播域时，就可以说成是把一个 LAN 划分成了 4 个 VLAN。

特别需要说明的是，在一个广播域内，任何两台终端计算机之间都可以进行二层（数据链路层）通信。所谓二层通信，是指通信的双方是以直接交换帧的方式来传递信息的。也就是说，目的计算机所接收到的帧与源计算机发出的帧是一模一样的，帧的目的 MAC 地址、源 MAC 地址、类型值、载荷数据、CRC 等内容都没有发生任何改变。二层通信方式中，信息源发送的帧可能会通过交换机进行二层转发，但一定不会经过路由器（或具有三层转发功能的交换机）进行三层转发。

源计算机在向目的计算机传递信息时，如果源计算机发出的帧经过了路由器（或具有三层转发功能的交换机）的转发，那么目的计算机接收到的帧一定不再是源计算机发出的那个帧。至少，目的计算机接收到的帧的目的 MAC 地址和源 MAC 地址一定不同于源计算机发出的帧的目的 MAC 地址和源 MAC 地址。在这样的情况下，源计算机与目的计算机之间的通信就不再是二层通信，而只能称为三层通信。

一个 VLAN 就是一个广播域，所以在同一个 VLAN 内部，计算机之间的通信就是二层通信。如果源计算机与目的计算机位于不同的 VLAN 中，那么它们之间是无法进行二层通信的，只能进行三层通信来传递信息。

5.2 VLAN 的基本原理

如图 5-3 所示，一台交换机连接了 6 台计算机，本来只是一个广播域，现在通过 VLAN 技术将之划分成了两个较小的广播域，分别是 VLAN 2 和 VLAN 3。由于在交换机上进行了相关的 VLAN 配置，所以交换机知道了自己的 Port 1、Port 2、Port 6 属于 VLAN 2，Port 3、Port 4、Port 5 属于 VLAN 3。注意，计算机本身是不能感知 VLAN 的，在计算机的“头脑”中完全没有 VLAN 的概念，在计算机上也不会进行任何有关 VLAN 的配置。

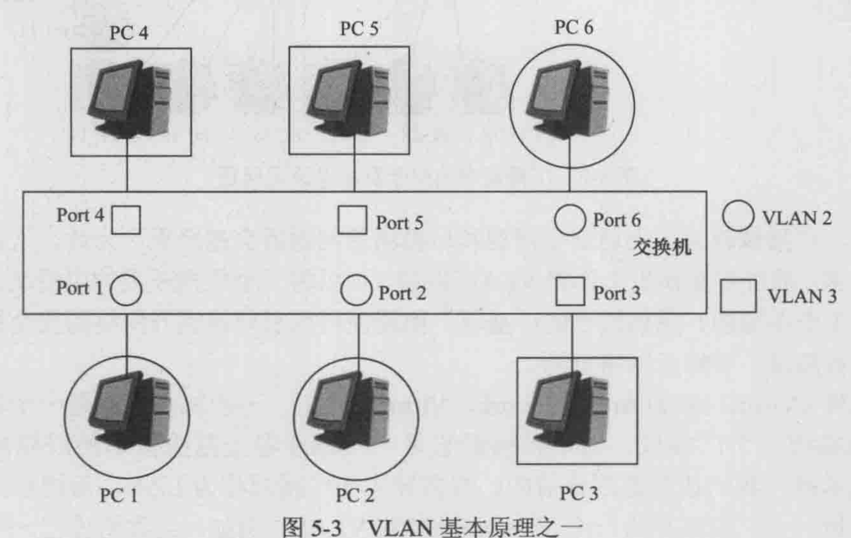


图 5-3 VLAN 基本原理之一

如图 5-4 所示，假设 PC 1 发送了一个广播帧 X。因为 X 帧是从属于 VLAN 2 的 Port 1

进入交换机的，所以交换机会判定 X 帧属于 VLAN 2，于是只会向同属于 VLAN 2 的 Port 2 和 Port 6 进行泛洪。最后，只有 PC 2 和 PC 6 能接收到 X 帧，而属于 VLAN 3 的 PC 3、PC 4、PC 5 是接收不到 X 帧的。

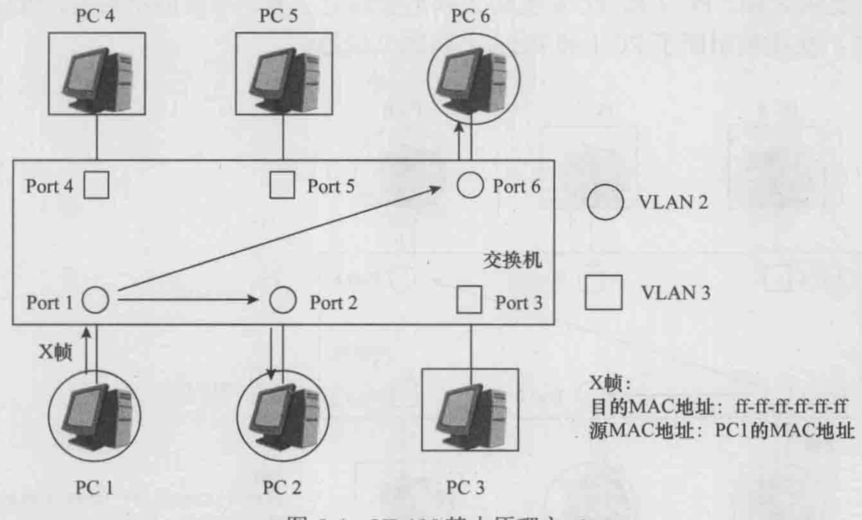


图 5-4　VLAN 基本原理之二

如图 5-5 所示，假设 PC 1 向 PC 6 发送了一个单播帧 Y，另假设交换机的 VLAN 2 的 MAC 地址表中存在关于 PC 6 的 MAC 地址的表项。因为 Y 帧是从属于 VLAN 2 的 Port 1 进入交换机的，所以交换机会判定 Y 帧属于 VLAN 2。交换机在查询了 VLAN 2 的 MAC 地址表后，会将 Y 帧点到点地向同属于 VLAN 2 的 Port 6 进行转发。最后，PC 6 便成功地接收到了 Y 帧（补充说明一下，如果交换机的 VLAN 2 的 MAC 地址表中不存在关于 PC 6 的 MAC 地址的表项，那么交换机会向 Port 2 和 Port 6 泛洪 Y 帧。PC 2 收到 Y 帧后会将之丢弃；PC 6 收到 Y 帧后不会将之丢弃，而是进行后续处理）。

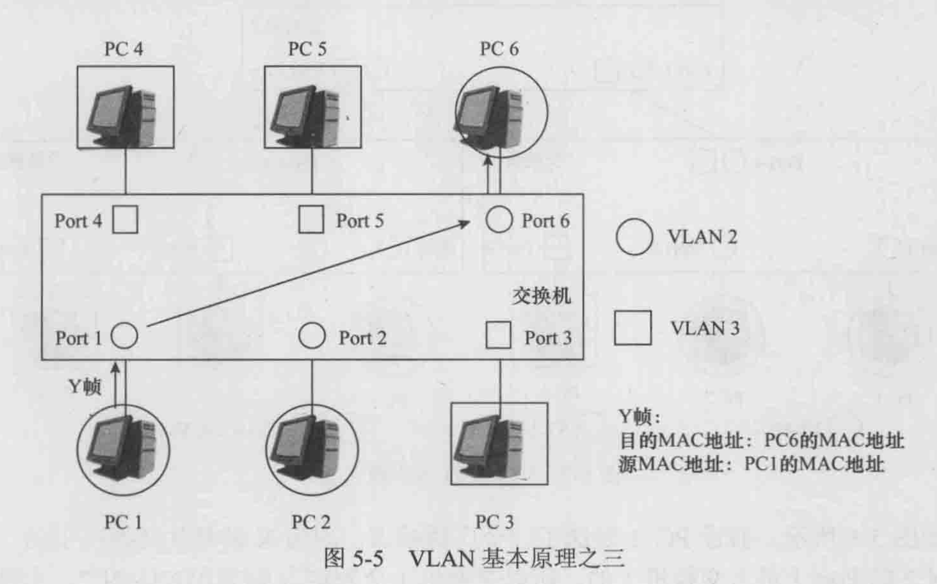


图 5-5　VLAN 基本原理之三

如图 5-6 所示，假设 PC 1 向 PC 3 发送了一个单播帧 Z。因为 Z 帧是从属于 VLAN 2 的 Port 1 进入交换机的，所以交换机会判定 Z 帧属于 VLAN 2。交换机的 VLAN 2 的 MAC 地址表中在正常情况下是不存在关于 PC 3 的 MAC 地址的表项的，所以交换机会向 Port 2 和 Port 6 泛洪 Z 帧。PC 2 和 PC 6 收到 Z 帧后会将之丢弃。最后的结果是，PC 3 无法接收到 Z 帧，交换机阻断了 PC 1 和 PC 3 之间的二层通信。

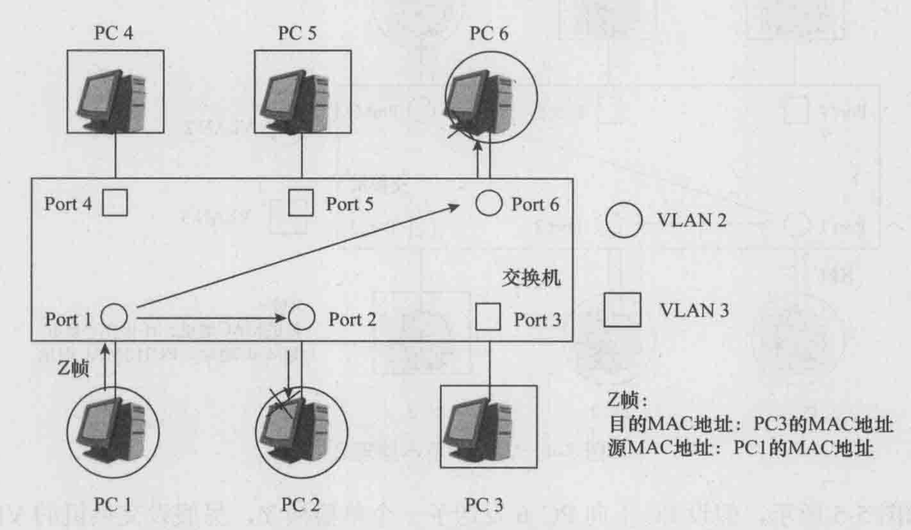


图 5-6 VLAN 基本原理之四

我们再来看一个比较复杂的例子。如图 5-7 所示，3 台交换机和 6 台计算机组成了一个交换网络，该网络现在被划分成了两个 VLAN，分别是 VLAN 2 和 VLAN 3。由于在每台交换机上都进行了 VLAN 配置，所以交换机知道自己的哪些端口属于 VLAN 2，哪些端口属于 VLAN 3，哪些端口既属于 VLAN 2，又属于 VLAN 3。

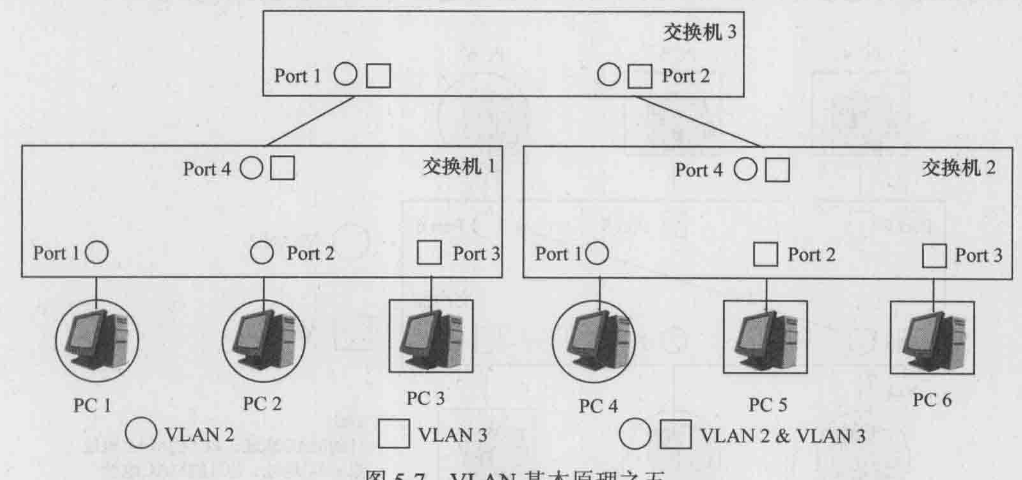


图 5-7 VLAN 基本原理之五

如图 5-8 所示，假设 PC 1 发送了一个广播帧 X。因为 X 帧是从交换机 1 的、属于 VLAN 2 的 Port 1 进入交换机 1 的，所以交换机 1 会判定 X 帧属于 VLAN 2，于是会向

Port 2 和 Port 4 进行泛洪。交换机 3 从其 Port 1 收到 X 帧后，会通过某种方法识别出 X 帧是属于 VLAN 2 的，于是会向 Port 2 泛洪 X 帧。交换机 2 从其 Port 4 收到 X 帧后，也会通过某种方法识别出 X 帧是属于 VLAN 2 的，于是会向 Port 1 泛洪 X 帧。最后，PC 2 和 PC 4 都会接收到 X 帧。

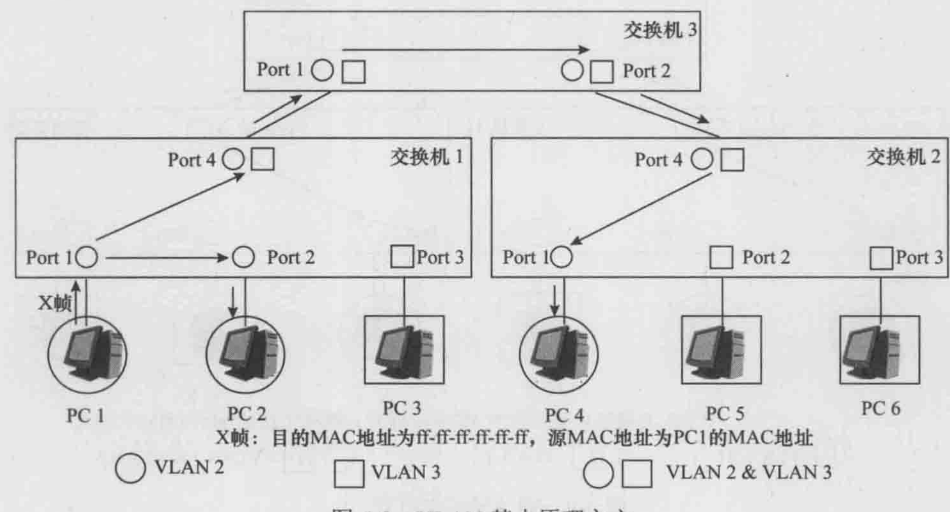


图 5-8 VLAN 基本原理之六

如图 5-9 所示，假设 PC 1 向 PC 4 发送了一个单播帧 Y，另假设所有交换机的 VLAN 2 的 MAC 地址表中都存在关于 PC 4 的 MAC 地址的表项。因为 Y 帧是从交换机 1 的、属于 VLAN 2 的 Port 1 进入交换机 1 的，所以交换机 1 会判定 Y 帧属于 VLAN 2。交换机 1 在查询了自己的 VLAN 2 的 MAC 地址表后，会将 Y 帧点到点地向 Port 4 进行转发。交换机 3 从其 Port 1 收到 Y 帧后，会通过某种方法识别出 Y 帧是属于 VLAN 2 的。交换机 3 在查询了自己的 VLAN 2 的 MAC 地址表后，会将 Y 帧点到点地向 Port 2 进行转发。交换机 2 从其 Port 4 收到 Y 帧后，也会通过某种方法识别出 Y 帧是属于 VLAN 2 的。交换机 2 在查询了自己的 VLAN 2 的 MAC 地址表后，会将 Y 帧点到点地向 Port 1 进行转发。最后，PC 4 便会接收到 Y 帧。顺便提一下，如果有的交换机的 VLAN 2 的 MAC 地址表中存在关于 PC 4 的 MAC 地址的表项，有的交换机的 VLAN 2 的 MAC 地址表中不存在关于 PC 4 的 MAC 地址的表项，那么情况会怎样呢？请读者朋友们自己去推断一下。

如图 5-10 所示，假设 PC 1 向 PC 6 发送了一个单播帧 Z。所有交换机的 VLAN 2 的 MAC 地址表中在正常情况下是不存在关于 PC 6 的 MAC 地址的表项的。因为 Z 帧是从交换机 1 的、属于 VLAN 2 的 Port 1 进入交换机 1 的，所以交换机 1 会判定 Z 帧属于 VLAN 2。交换机 1 在自己的 VLAN 2 的 MAC 地址表中查不到关于 PC 6 的 MAC 地址的表项，所以交换机 1 会向 Port 2 和 Port 4 泛洪 Z 帧。交换机 3 从其 Port 1 收到 Z 帧后，会通过某种方法识别出 Z 帧是属于 VLAN 2 的。交换机 3 在自己的 VLAN 2 的 MAC 地址表中查不到关于 PC 6 的 MAC 地址的表项，所以交换机 3 会向 Port 2 泛洪 Z 帧。交换机 2 从其 Port 4 收到 Z 帧后，也会通过某种方法识别出 Z 帧是属于 VLAN 2 的。交换机 2 在自己的 VLAN 2 的 MAC 地址表中查不到关于 PC 6 的 MAC 地址的表

项，所以交换机 2 会向 Port 1 泛洪 Z 帧。最后，PC 2 和 PC 4 都会接收到 Z 帧，但都会将之丢弃。PC 6 并不能接收到 PC 1 发送给自己的 Z 帧，交换机阻断了 PC 1 和 PC 6 之间的二层通信。

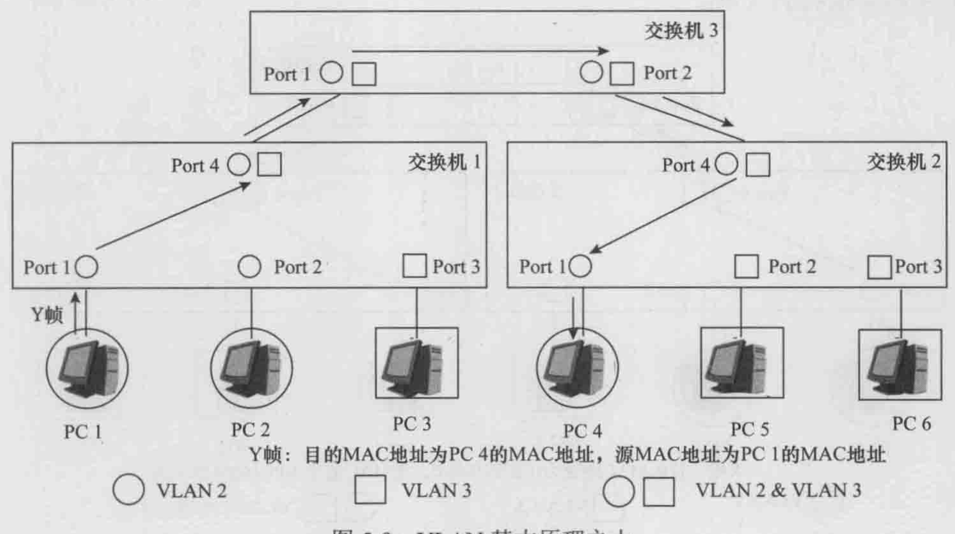


图 5-9　VLAN 基本原理之七

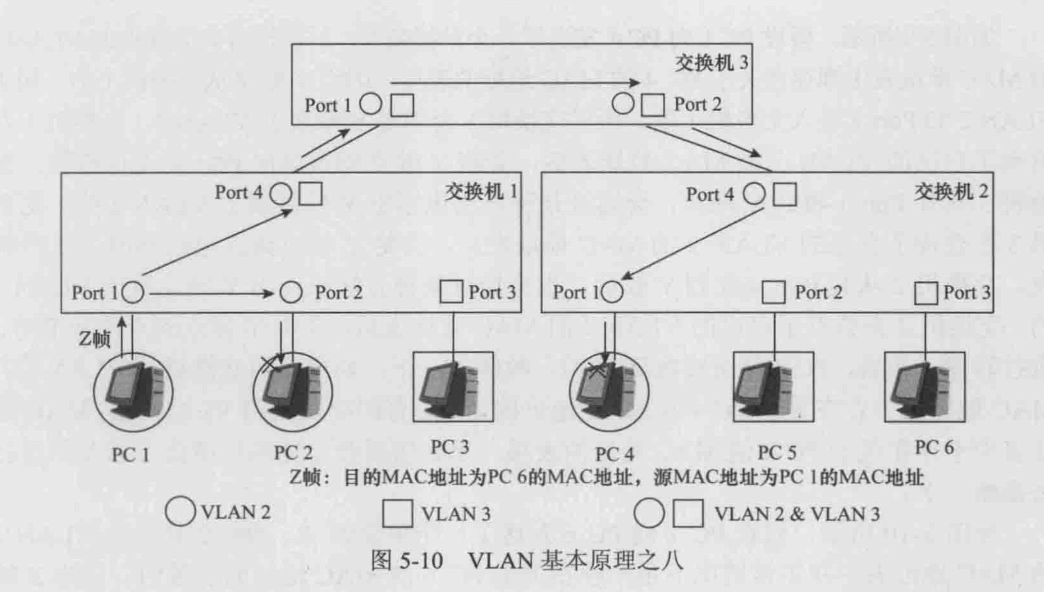


图 5-10　VLAN 基本原理之八

5.3　802.1Q 帧的格式

IEEE 802.1D 定义了关于不支持 VLAN 特性的交换机的标准规范，IEEE 802.1Q 定义了关于支持 VLAN 特性的交换机的标准规范。IEEE 802.1Q 的内容覆盖了 IEEE 802.1D 的所有内容，并增加了有关 VLAN 特性的内容。IEEE 802.1Q 可以后向兼容 IEEE 802.1D。如无特

别说明，接下来提到的交换机都是指能够支持 VLAN 特性的、遵从 802.1Q 标准的交换机。

交换机在识别一个帧是属于哪个 VLAN 的时候，可以根据这个帧是从哪个端口进入自己的来进行判定，也可能需要根据别的信息来进行判定。通常，交换机识别出某个帧是属于哪个 VLAN 后，会在这个帧的特定位置上添加上一个标签（Tag），这个 Tag 明确地表明了这个帧是属于哪个 VLAN 的。这样一来，别的交换机收到这个带 Tag 的帧后，就能轻易而举地直接根据 Tag 信息识别出这个帧是属于哪个 VLAN 的。IEEE 802.1Q 定义了这种带 Tag 的帧的格式，满足这种格式的帧称为 IEEE 802.1Q 帧，如图 5-11 所示。

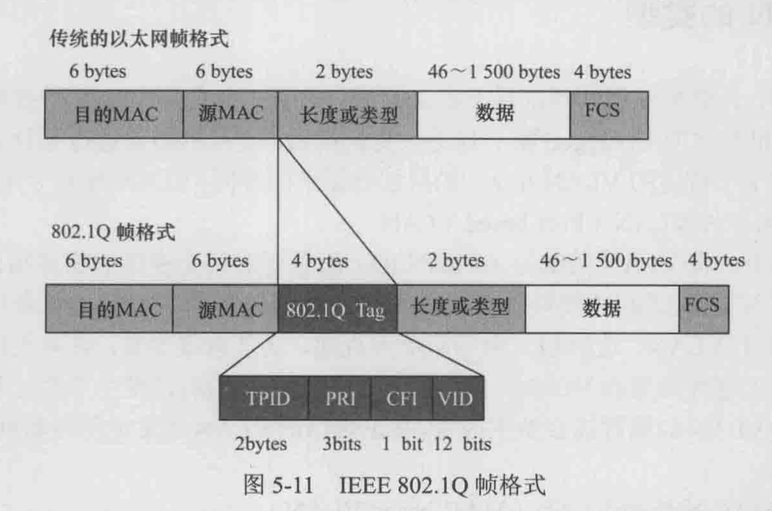


图 5-11 IEEE 802.1Q 帧格式

关于 IEEE 802.1Q 帧格式中 802.1Q Tag（也称为 VLAN Tag）的各个字段的含义如表 5-1 所示。

表 5-1 802.1Q Tag 中各个字段的含义

字段	长度	名称	解释
TPID	2 bytes	Tag Protocol ID，表示这个帧是否带有 Tag	如果 TPID 取值为 0x8100，则表示该帧是带 Tag 的帧；否则表示该帧是传统的、不带 Tag 的帧。该字段与传统以太网帧中该位置的 Type/Length 字段是兼容的
PRI	3 bits	Priority，表示帧的优先级	取值范围为 0～7，值越大优先级越高。用于当交换机阻塞时，先发送优先级较高的帧
CFI	1 bit	Canonical Format Indicator	其含义已超出了本书的知识范围，在此不作解释
VID	12 bits	VLAN Identifier，表示该帧所属的 VLAN	VLAN ID 取值范围是 0～4 095。由于 0 和 4 095 为协议保留取值，所以 VLAN ID 的有效取值范围是 1～4 094

从图 5-11 和表 5-1 我们可以看出，如果一个帧的源 MAC 地址后面的两个字节的值

是 0x8100，则说明这个帧是一个 Tagged 帧；如果一个帧的源 MAC 地址后面的两个字节的值不是 0x8100，则说明这个帧是一个传统的 Untagged 帧。

另外，需要再次指出的是，计算机的“头脑”中是没有任何关于 VLAN 的概念的。计算机不会产生并发送 Tagged 帧。如果计算机接收到了一个 Tagged 帧，由于它识别不出 0x8100 的含义，于是会直接将这个 Tagged 帧丢弃。

5.4 VLAN 的类型

刚才说到，计算机发送的帧都是不带 Tag 的。对于一个支持 VLAN 特性的交换网络来说，当计算机发送的 Untagged 帧一旦进入交换机后，交换机必须通过某种划分原则把这个帧划分到某个特定的 VLAN 中去。根据划分原则的不同，VLAN 便有了不同的类型。

1. 基于端口的 VLAN（Port-based VLAN）

其划分原则：将 VLAN 的编号（VLAN ID）配置影射到交换机的物理端口上，从某一物理端口进入交换机的、由终端计算机发送的 Untagged 帧都被划分到该端口的 VLAN ID 所表明的那个 VLAN。这种划分原则简单而直观，实现也很容易，并且也比较安全可靠。注意，对于这种类型的 VLAN，当计算机接入交换机的端口发生了变化时，该计算机发送的帧的 VLAN 归属可能会发生改变。基于端口的 VLAN 通常也称为物理层 VLAN，或一层 VLAN。

2. 基于 MAC 地址的 VLAN（MAC-based VLAN）

其划分原则：交换机内部建立并维护了一个 MAC 地址与 VLAN ID 的对应表，当交换机接收到计算机发送的 Untagged 帧时，交换机将分析帧中的源 MAC 地址，然后查询 MAC 地址与 VLAN ID 的对应表，并根据对应关系把这个帧划分到相应的 VLAN 中。这种划分原则实现起来稍显复杂，但灵活性得到了提高。例如，当计算机接入交换机的端口发生了变化时，该计算机发送的帧的 VLAN 归属并不会发生改变（因为计算机的 MAC 地址不会发生变化）。但需要指出的是，这种类型的 VLAN 的安全性不是很高，因为一些恶意的计算机是很容易伪造自己的 MAC 地址的。基于 MAC 地址的 VLAN 通常也称为二层 VLAN。

3. 基于协议的 VLAN（Protocol-based VLAN）

其划分原则：交换机根据计算机发送的 Untagged 帧中的帧类型字段的值来决定帧的 VLAN 归属。例如，可以将类型值为 0x0800 的帧划分到一个 VLAN，将类型值为 0x86dd 的帧划分到另一个 VLAN；这实际上是将载荷数据为 IPv4 Packet 的帧和载荷数据为 IPv6 Packet 的帧分别划分到了不同的 VlAN。基于协议的 VLAN 通常也称为三层 VLAN。

以上介绍了 3 种不同类型的 VLAN。从理论上说，VLAN 的类型远远不止这些，因为划分 VLAN 的原则可以是灵活而多变的，并且某一种划分原则还可以是另外若干种划分原则的某种组合。在现实中，究竟该选择什么样的划分原则，需要根据网络的具体需求、实现成本等因素决定。就目前来看，基于端口的 VLAN 在实际的网络中应用最为广泛。如无特别说明，本书中所提到的 VLAN，均是指基于端口的 VLAN。

5.5 链路类型和端口类型

一个 VLAN 帧可能带有 Tag（称为 Tagged VLAN 帧，或简称为 Tagged 帧），也可能不带 Tag（称为 Untagged VLAN 帧，或简称为 Untagged 帧）。在谈及 VLAN 技术时，如果一个帧被交换机划分到 VLAN *i*（*i* = 1，2，3，……，4 094），我们就把这个帧简称为一个 VLAN *i* 的帧，或一个 VLAN *i* 帧。对于带有 Tag 的 VLAN *i* 帧，*i* 其实就是这个帧的 Tag 中的 VID 字段的取值（见表 5-1）。注意，对于 Tagged VLAN 帧，交换机显然能够从其 Tag 中的 VID 值判定出它属于哪个 VLAN；对于 Untagged VLAN 帧（例如终端计算机发出的帧），交换机需要根据某种原则（比如根据这个帧是从哪个端口进入交换机的）来判定或划分它属于哪个 VLAN。

在一个支持 VLAN 特性的交换网络中，我们把交换机与终端计算机直接相连的链路称为 Access 链路（Access Link），把 Access 链路上交换机一侧的端口称为 Access 端口（Access Port）。同时，我们把交换机之间直接相连的链路称为 Trunk 链路（Trunk Link），把 Trunk 链路上两侧的端口称为 Trunk 端口（Trunk Port）。在一条 Access 链路上运动的帧只能是（或者说应该是）Untagged 帧，并且这些帧只能属于某个特定的 VLAN；在一条 Trunk 链路上运动的帧只能是（或者说应该是）Tagged 帧，并且这些帧可以属于不同的 VLAN。一个 Access 端口只能属于某个特定的 VLAN，并且只能让属于这个特定 VLAN 的帧通过；一个 Trunk 端口可以同时属于多个 VLAN，并且可以让属于不同 VLAN 的帧通过，如图 5-12 所示。

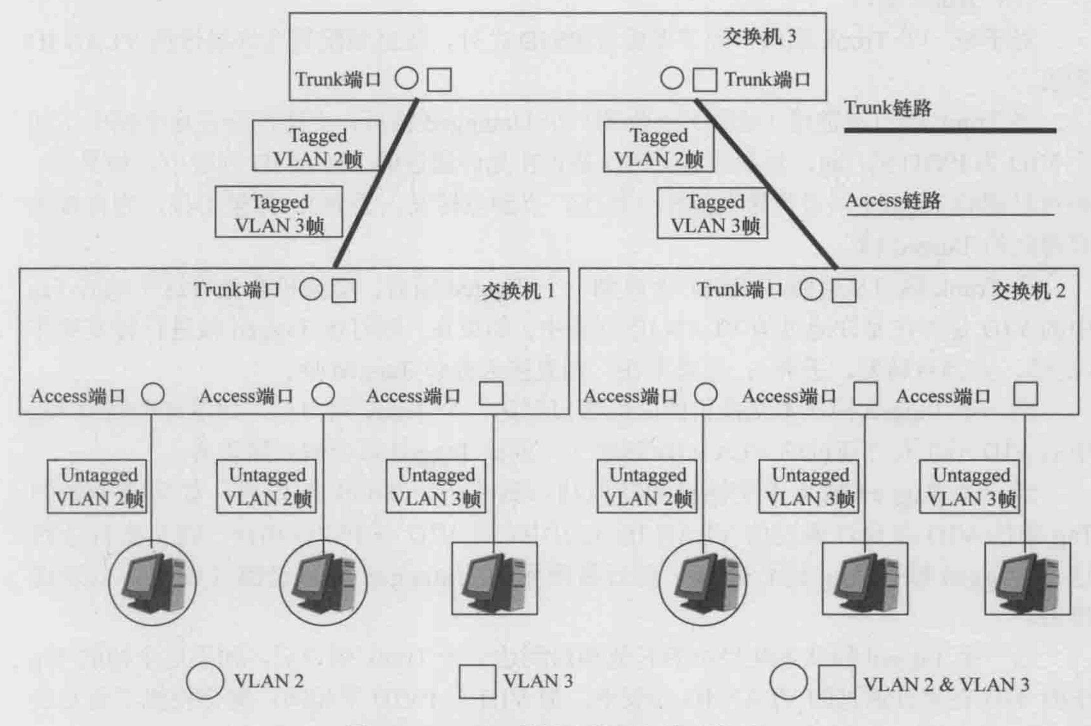


图 5-12 VLAN 的链路类型和端口类型

每一个交换机的端口（无论是 Access 端口还是 Trunk 端口）都应该配置一个 PVID（Port VLAN ID），到达这个端口的 Untagged 帧将一律被交换机划分到 PVID 所指代的 VLAN。例如，如果一个端口的 PVID 被配置为 5，则所有到达这个端口的 Untagged 帧都将被认定为是属于 VLAN 5 的帧。默认情况下，PVID 的值为 1。

概括地讲，链路（线路）上运动的帧，可能是 Tagged 帧，也可能是 Untagged 帧。但一台交换机内部不同端口之间运动的帧则一定是（或者说应该是）Tagged 帧。

接下来，我们具体地描述一下 Access 端口和 Trunk 端口对于帧的处理和转发规则。

（1）Access 端口

当 Access 端口从链路（线路）上收到一个 Untagged 帧后，交换机会在这个帧中添加上 VID 为 PVID 的 Tag，然后对得到的 Tagged 帧进行转发操作（泛洪，点到点转发，丢弃）。

当 Access 端口从链路（线路）上收到一个 Tagged 帧后，交换机会检查这个帧的 Tag 中的 VID 是否与 PVID 相同。如果相同，则对这个 Tagged 帧进行转发操作（泛洪，点到点转发，丢弃）；如果不同，则直接丢弃这个 Tagged 帧。

当一个 Tagged 帧从本交换机的其他端口到达一个 Access 端口后，交换机会检查这个帧的 Tag 中的 VID 是否与 PVID 相同。如果相同，则将这个 Tagged 帧的 Tag 进行剥离，然后将得到的 Untagged 帧从链路（线路）上发送出去；如果不同，则直接丢弃这个 Tagged 帧。

（2）Trunk 端口

对于每一个 Trunk 端口，除了要配置 PVID 之外，还必须配置允许通过的 VLAN ID 列表。

当 Trunk 端口从链路（线路）上收到一个 Untagged 帧后，交换机会在这个帧中添加上 VID 为 PVID 的 Tag，然后查看 PVID 是否在允许通过的 VLAN ID 列表中。如果在，则对得到的 Tagged 帧进行转发操作（泛洪，点到点转发，丢弃）；如果不在，则直接丢弃得到的 Tagged 帧。

当 Trunk 端口从链路（线路）上收到一个 Tagged 帧后，交换机会查看这个帧的 Tag 中的 VID 是否在允许通过的 VLAN ID 列表中。如果在，则对该 Tagged 帧进行转发操作（泛洪，点到点转发，丢弃）；如果不在，则直接丢弃该 Tagged 帧。

当一个 Tagged 帧从本交换机的其他端口到达一个 Trunk 端口后，如果这个帧的 Tag 中的 VID 不在允许通过的 VLAN ID 列表中，则该 Tagged 帧会被直接丢弃。

当一个 Tagged 帧从本交换机的其他端口到达一个 Trunk 端口后，如果这个帧的 Tag 中的 VID 在允许通过的 VLAN ID 列表中，且 VID 与 PVID 相同，则交换机会对这个 Tagged 帧的 Tag 进行剥离，然后将得到的 Untagged 帧从链路（线路）上发送出去。

当一个 Tagged 帧从本交换机的其他端口到达一个 Trunk 端口后，如果这个帧的 Tag 中的 VID 在允许通过的 VLAN ID 列表中，但 VID 与 PVID 不相同，则交换机不会对这个 Tagged 帧的 Tag 进行剥离，而是直接将它从链路（线路）上发送出去。

以上是对 Access 端口和 Trunk 端口的工作机制的描述。在实际的 VLAN 技术实现

中，还常常会定义并配置另外一种类型的端口，称为 Hybrid 端口，既可以将交换机上与终端计算机相连的端口配置为 Hybrid 端口，也可以将交换机上与其他交换机相连的端口配置为 Hybrid 端口。

（3）Hybrid 端口

Hybrid端口除了需要配置PVID外，还需要配置两个VLAN ID列表，一个是Untagged VLAN ID 列表，另一个是 Tagged VLAN ID 列表。这两个 VLAN ID 列表中的所有 VLAN 的帧都是允许通过这个 Hybrid 端口的。

当 Hybrid 端口从链路（线路）上收到一个 Untagged 帧后，交换机会在这个帧中添加上 VID 为 PVID 的 Tag，然后查看 PVID 是否在 Untagged VLAN ID 列表或 Tagged VLAN ID 列表中。如果在，则对得到的 Tagged 帧进行转发操作（泛洪，点到点转发，丢弃）；如果不在，则直接丢弃得到的 Tagged 帧。

当 Hybrid 端口从链路（线路）上收到一个 Tagged 帧后，交换机会查看这个帧的 Tag 中的 VID 是否在 Untagged VLAN ID 列表或 Tagged VLAN ID 列表中。如果在，则对该 Tagged 帧进行转发操作（泛洪，点到点转发，丢弃）；如果不在，则直接丢弃该 Tagged 帧。

当一个 Tagged 帧从本交换机的其他端口到达一个 Hybrid 端口后，如果这个帧的 Tag 中的 VID 既不在 Untagged VLAN ID 列表中，也不在 Tagged VLAN ID 列表中，则该 Tagged 帧会被直接丢弃。

当一个 Tagged 帧从本交换机的其他端口到达一个 Hybrid 端口后，如果这个帧的 Tag 中的 VID 在 Untagged VLAN ID 列表中，则交换机会对这个 Tagged 帧的 Tag 进行剥离，然后将得到的 Untagged 帧从链路（线路）上发送出去。

当一个 Tagged 帧从本交换机的其他端口到达一个 Hybrid 端口后，如果这个帧的 Tag 中的 VID 在 Tagged VLAN ID 列表中，则交换机不会对这个 Tagged 帧的 Tag 进行剥离，而是直接将它从链路（线路）上发送出去。

Hybrid 端口的工作机制比 Trunk 端口和 Access 端口更为丰富而灵活：Trunk 端口和 Access 端口可以看成是 Hybrid 端口的特例。当 Hybrid 端口配置中的 Untagged VLAN ID 列表中有且只有 PVID 时，Hybrid 端口就等效于一个 Trunk 端口；当 Hybrid 端口配置中的 Untagged VLAN ID 列表中有且只有 PVID，并且 Tagged VLAN ID 列表为空时，Hybrid 端口就等效于一个 Access 端口。

5.6 VLAN 转发示例

如图 5-13 所示，假定交换机 1 的 Port 1 和 Port 2 以及交换机 2 的 Port 1 的 PVID 是 VLAN 2，假定交换机 1 的 Port 3 以及交换机 2 的 Port 2 和 Port 3 的 PVID 是 VLAN 3，假定所有 Trunk 端口的 PVID 是 VLAN 1，假定所有 Trunk 端口都允许 VLAN 2 和 VLAN 3 的帧通过，假设 PC 1 发送了一个 Untagged 广播帧 X，那么 X 帧从交换机 1 的 Port 1 进入交换机 1 后，交换机 1 会给 X 帧打上 VID 为 VLAN 2 的 Tag，然后向 Port 2 和 Port 4 进行泛洪；交换机 1 的 Port 2 收到来自 Port 1 的 Tagged X 帧后，会剥去 Tag，然后将

Untagged X 帧发送给 PC 2；交换机 1 的 Port 4 收到来自 Port 1 的 Tagged X 帧后，会直接发送给交换机 3 的 Port 1。交换机 3 会把从 Port 1 进入的 Tagged X 帧直接向 Port 2 泛洪；交换机 3 的 Port 2 会直接将来自 Port 1 的 Tagged X 帧发送给交换机 2 的 Port 4。交换机 2 会对进入 Port 4 的 Tagged X 帧直接向 Port 1 泛洪；交换机 2 的 Port 1 收到来自 Port 4 的 Tagged X 帧后，会剥去 Tag，然后将 Untagged X 帧发送给 PC 4。最后 PC 2 和 PC 4 都会接收到不带 Tag 的 X 帧。

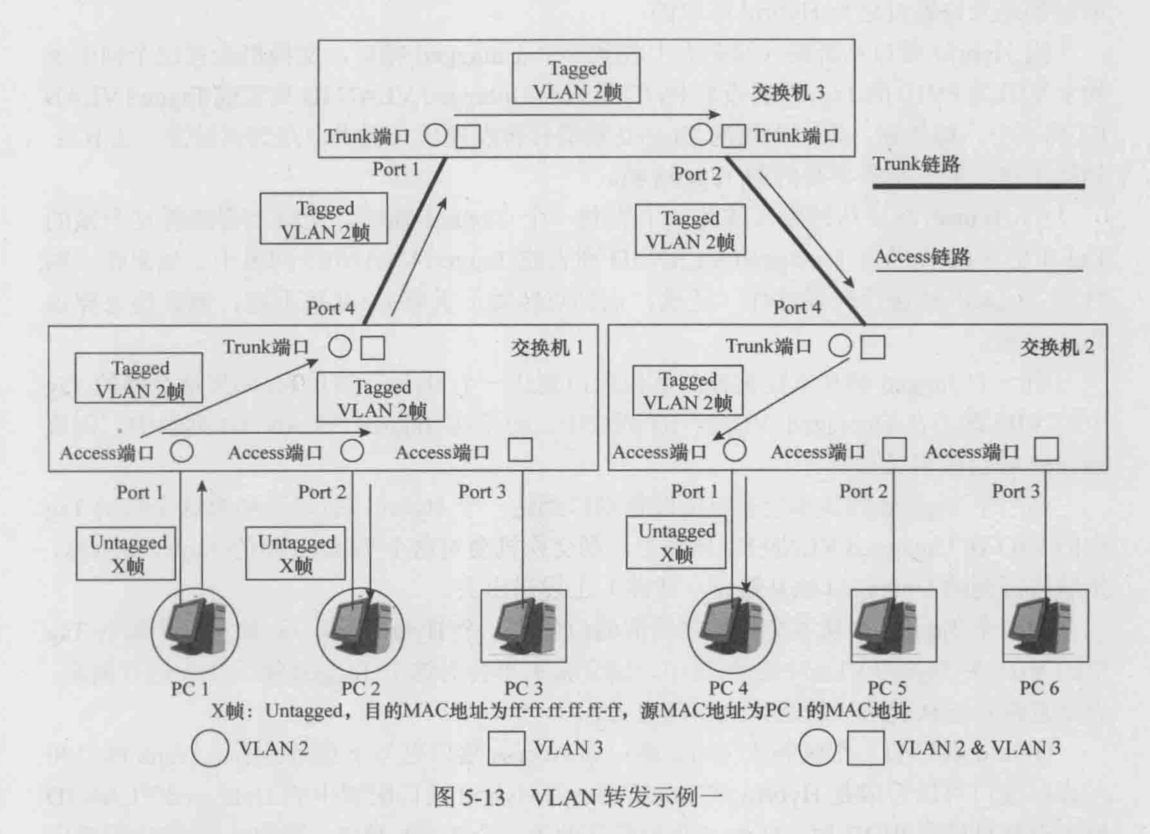


图 5-13 VLAN 转发示例一

如图 5-14 所示，假定交换机 1 的 Port 1 和 Port 2 以及交换机 2 的 Port 1 的 PVID 是 VLAN 2，假定交换机 1 的 Port 3 以及交换机 2 的 Port 2 和 Port 3 的 PVID 是 VLAN 3，假定所有 Trunk 端口的 PVID 是 VLAN 1，假定所有 Trunk 端口都允许 VLAN 2 和 VLAN 3 的帧通过，假定所有交换机的 VLAN 2 的 MAC 地址表中都存在关于 PC 4 的 MAC 地址的表项，假设 PC 1 向 PC 4 发送了一个 Untagged 单播帧 Y，那么 Y 帧从交换机 1 的 Port 1 进入交换机 1 后，交换机 1 会给 Y 帧打上 VID 为 VLAN 2 的 Tag；交换机 1 在查询了自己的 VLAN 2 的 MAC 地址表后，会将 Tagged Y 帧点到点地向 Port 4 进行转发；交换机 1 的 Port 4 收到来自 Port 1 的 Tagged Y 帧后，会直接发送给交换机 3 的 Port 1。交换机 3 从其 Port 1 收到 Tagged Y 帧后，查询自己的 VLAN 2 的 MAC 地址表，然后将 Tagged Y 帧点到点地向 Port 2 进行转发；交换机 3 的 Port 2 会直接将来自 Port 1 的 Tagged Y 帧发送给交换机 2 的 Port 4。交换机 2 从其 Port 4 收到 Tagged Y 帧后，查询自己的 VLAN 2 的 MAC 地址表，然后

将 Tagged Y 帧点到点地向 Port 1 进行转发；交换机 2 的 Port 1 收到来自 Port 4 的 Tagged Y 帧后，会剥去 Tag，然后将 Untagged Y 帧发送给 PC 4。最后，PC 4 便会接收到不带 Tag 的 Y 帧。顺便提一下，如果有的交换机的 VLAN 2 的 MAC 地址表中存在关于 PC 4 的 MAC 地址的表项，有的交换机的 VLAN 2 的 MAC 地址表中不存在关于 PC 4 的 MAC 地址的表项，那么情况会怎样呢？请读者朋友们自己去推断一下。

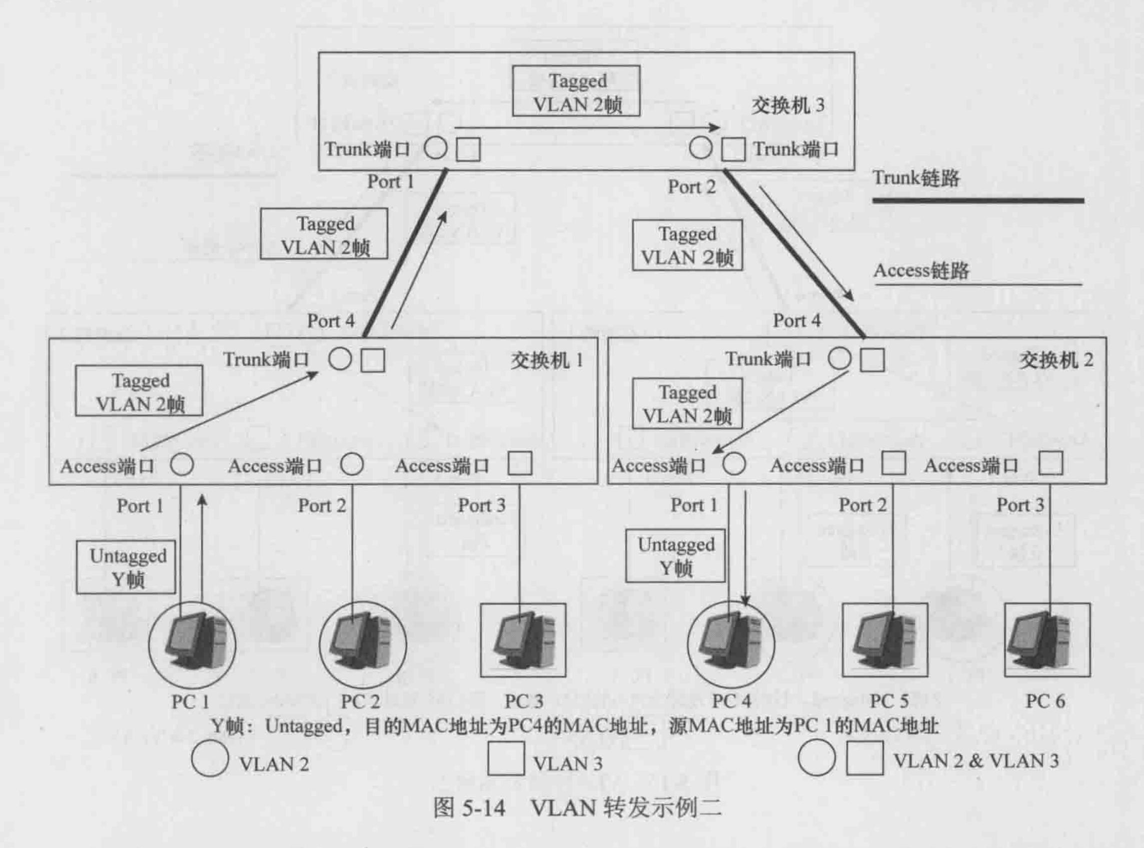


图 5-14 VLAN 转发示例二

如图 5-15 所示，假定交换机 1 的 Port 1 和 Port 2 以及交换机 2 的 Port 1 的 PVID 是 VLAN 2，假定交换机 1 的 Port 3 以及交换机 2 的 Port 2 和 Port 3 的 PVID 是 VLAN 3，假定所有 Trunk 端口的 PVID 是 VLAN 1，假定所有 Trunk 端口都允许 VLAN 2 和 VLAN 3 的帧通过。注意，所有交换机的 VLAN 2 的 MAC 地址表中在正常情况下是不存在关于 PC 6 的 MAC 地址的表项的。假设 PC 1 向 PC 6 发送了一个 Untagged 单播帧 Z。Z 帧从交换机 1 的 Port 1 进入交换机 1 后，交换机 1 会给 Z 帧打上 VID 为 VLAN 2 的 Tag。交换机 1 在自己的 VLAN 2 的 MAC 地址表中查不到关于 PC 6 的 MAC 地址的表项，所以交换机 1 会向 Port 2 和 Port 4 泛洪 Tagged Z 帧。交换机 1 的 Port 2 收到来自 Port 1 的 Tagged Z 帧后，会剥去 Tag，然后将 Untagged Z 帧发送给 PC 2。交换机 1 的 Port 4 收到来自 Port 1 的 Tagged Z 帧后，会直接发送给交换机 3 的 Port 1。交换机 3 从其 Port 1 收到 Tagged Z 帧后，在自己的 VLAN 2 的 MAC 地址表中查不到关于 PC 6 的 MAC 地址的表项，所以交换机 3 会向 Port 2 泛洪 Tagged Z 帧。交换机 3 的 Port 2 会直接将来自 Port 1 的 Tagged Z

帧发送给交换机 2 的 Port 4。交换机 2 从其 Port 4 收到 Tagged Z 帧后，在自己的 VLAN 2 的 MAC 地址表中查不到关于 PC 6 的 MAC 地址的表项，所以交换机 2 会向 Port 1 泛洪 Tagged Z 帧。交换机 2 的 Port 1 收到来自 Port 4 的 Tagged Z 帧后，会剥去 Tag，然后将 Untagged Z 帧发送给 PC 4。最后，PC 2 和 PC 4 都会接收到不带 Tag 的 Z 帧，但都会将之丢弃。PC 6 并不能接收到 PC 1 发送给自己的 Z 帧，交换机阻断了 PC 1 和 PC 6 之间的二层通信。

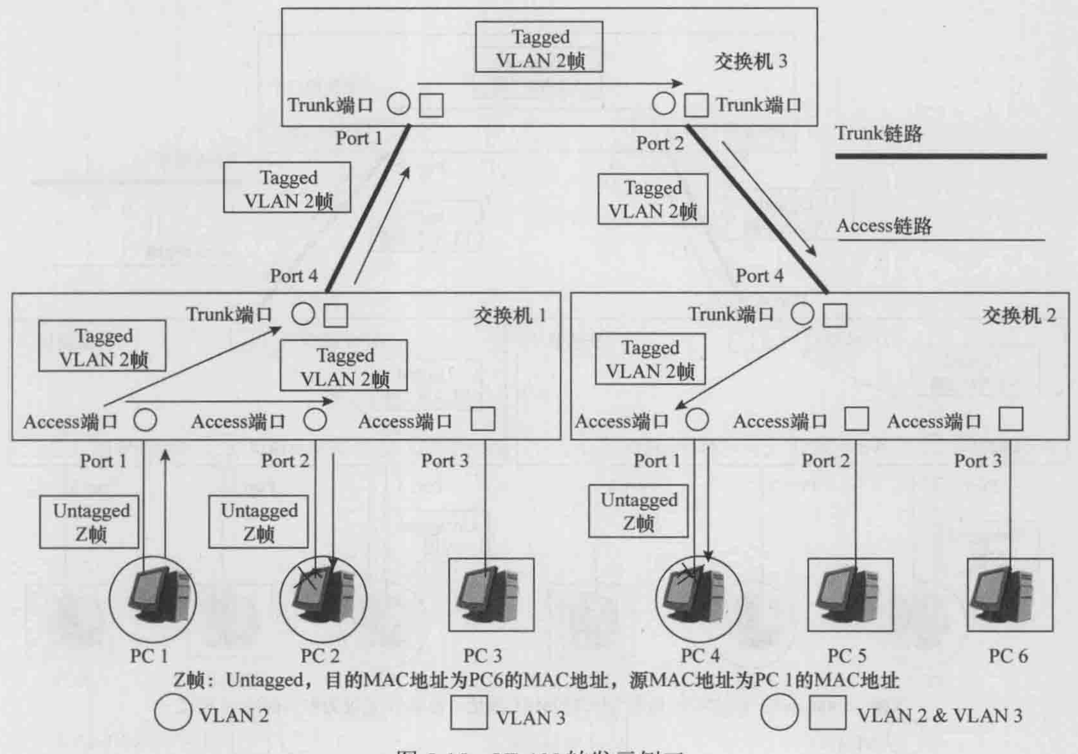


图 5-15　VLAN 转发示例三

5.7 VLAN 配置示例

某公司的交换网络如图 5-16 所示，其中 PC 1 和 PC 2 同属一个部门，PC 3、PC 4、PC 5 同属一个部门，PC 6 单独属于一个部门。为了阻断不同部门之间的二层通信，划分了 3 个 VLAN，分别为 VLAN 10、VLAN 20、VLAN 30。

1. 配置思路

（1）在交换机上创建 VLAN。

（2）配置交换机上连接 PC 的端口为 Access 模式，并加入相应的 VLAN。

（3）配置交换机之间互连的端口为 Trunk 模式，并加入相应的 VLAN。

2. 配置步骤

下面只针对 VLAN 10 给出了具体的配置步骤。

要在交换机上配置 VLAN，必须首先进入系统视图。然后，执行命令 **vlan** *vlan-id*，创建 VLAN。

#配置 S2。

```
<S2> system-view
[S2] vlan 10
[S2-vlan10] quit
```

#配置 S3。

```
<S3> system-view
[S3] vlan 10
[S3-vlan10] quit
```

#配置 S1。

```
<S1> system-view
[S1] vlan 10
[S1-vlan10] quit
```

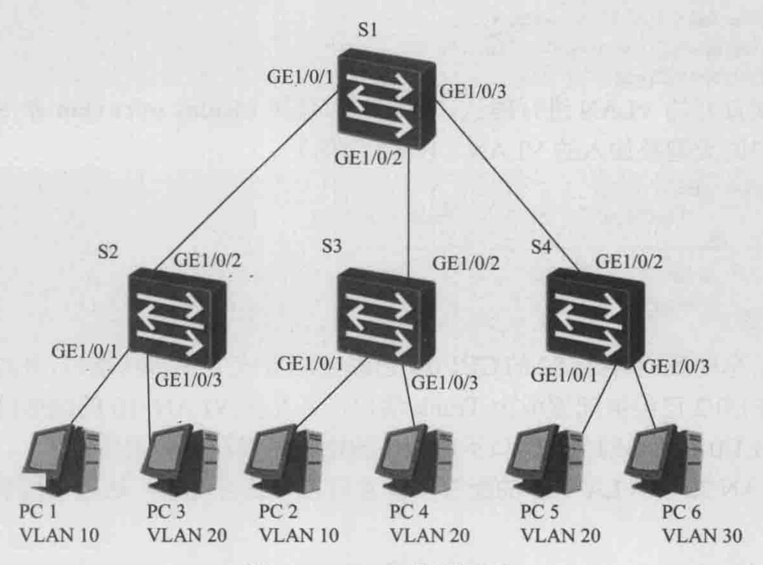


图 5-16 VLAN 配置示例

在创建了 VLAN 10 之后，VLAN 10 里并没有任何端口。因此，我们还需要将端口和对应的 VLAN 关联起来。

交换机的端口类型缺省为 Hybrid，使用命令 **port link-type{access|trunk}** 可将端口类型修改成为 Access 端口或 Trunk 端口。对于 Access 端口，配置其 PVID 的命令是 **port default vlan** *vlan-id*。对于 Trunk 端口，命令 **port trunk allow-pass vlan** *vlan-id1*[**to** *vlan-id2*]可以用来配置端口所允许通过的 VLAN。

#配置 S2。

```
[S2] interface gigabitethernet 1/0/1
[S2-GigabitEthernet1/0/1] port link-type access
[S2-GigabitEthernet1/0/1] port default vlan 10
[S2-GigabitEthernet1/0/1] quit
[S2] interface gigabitethernet 1/0/2
[S2-GigabitEthernet1/0/2] port link-type trunk
```

```
[S2-GigabitEthernet1/0/2] port trunk allow-pass vlan 10
[S2-GigabitEthernet1/0/2] quit
```

#配置 S3。

```
[S3] interface gigabitethernet 1/0/1
[S3-GigabitEthernet1/0/1] port link-type access
[S3-GigabitEthernet1/0/1] port default vlan 10
[S3-GigabitEthernet1/0/1] quit
[S3] interface gigabitethernet 1/0/2
[S3-GigabitEthernet1/0/2] port link-type trunk
[S3-GigabitEthernet1/0/2] port trunk allow-pass vlan 10
[S3-GigabitEthernet1/0/2] quit
```

#配置 S1。

```
[S1] interface gigabitethernet 1/0/1
[S1-GigabitEthernet1/0/1] port link-type trunk
[S1-GigabitEthernet1/0/1] port trunk allow-pass vlan 10
[S1-GigabitEthernet1/0/1] quit
[S1] interface gigabitethernet 1/0/2
[S1-GigabitEthernet1/0/2] port link-type trunk
[S1-GigabitEthernet1/0/2] port trunk allow-pass vlan 10
[S1-GigabitEthernet1/0/2] quit
```

为了对配置好的 VLAN 进行确认，我们可以使用 **display port vlan** 命令来查看交换机当前各端口的类型及加入的 VLAN（以 S2 为例）。

```
<S2> display port vlan
Port              Link Type     PVID     Trunk VLAN List
--------------------------------------------------------------------------
GE1/0/1           access        10       -
GE1/0/2           trunk         1        1 10
......
```

从回显信息中我们看到，S2 的 GE1/0/1 已经被配置成为 Access 端口，并加入了 VLAN 10；S2 的 GE1/0/2 已经被配置成为 Trunk 端口，并允许 VLAN 10 的帧通过。这表明我们在 S2 的 GE1/0/1 和 GE1/0/2 端口下所使用的命令已经在设备上生效了。

有关 VLAN 20 和 VLAN 30 的配置，读者可自行练习完成，这里不再赘述。

5.8 GVRP

在对交换机进行 VLAN 配置的时候，需要在每一台交换机上手工创建该交换机需要涉及到的所有 VLAN，并且需要手工将交换机上的各个端口加入到相应的 VLAN 中去。如果交换机及 VLAN 的数量太多，特别是 VLAN 的数量又经常变化时，则这种手工配置方式既耗费时间和精力，又非常容易出错。

为此，IEEE 制定了一个名为 GARP（Generic Attribute Registration Protocol，通用属性注册协议）的框架协议，该框架协议包含了两个具体的协议，分别称为 GMRP（GARP Multicast Registration Protocl，简称 GMRP）和 GVRP（GARP VLAN Registration Protocl，简称 GVRP）。GVRP 的应用，可以大大地降低 VLAN 配置过程中的手工工作量。

在涉及 GVRP 协议的应用时，我们通常将交换机上手工创建的 VLAN 称为静态

VLAN，而将交换机利用GVRP协议自动创建的VLAN称为动态VLAN。GVRP协议提供了一种在交换机之间传递VLAN属性的机制，其主要作用是自动实现VLAN信息在交换机上的动态注册过程和注销过程。交换机上部署了GVRP协议后，用户只需要对少量的交换机进行静态VLAN配置，便可将这些VLAN配置信息传递并应用到其他的交换机上。

5.8.1　VLAN属性的动态注册过程

如图5-17所示，PC 1和PC 2都被划分到VLAN 10，因此所有的交换机上都要进行针对VLAN 10的配置。假设交换机S1、S2、S3、S4都已经全局使能了GVRP功能，并且相关的端口（S1的GE0/0/1、S2的GE0/0/1和GE0/0/2、S3的GE0/0/1和GE0/0/2、S4的GE0/0/1）也都使能了GVRP功能，那么当用户在交换机S1上手工创建了静态VLAN 10，并且配置S1的GE0/0/1端口允许属于VLAN 10的帧通过之后，S1的GE0/0/1端口便会向外发送VLAN属性的注册报文。

S2通过其GE0/0/1端口接收到S1发送过来的VLAN属性注册报文后，会自动创建动态VLAN 10，并将自己的GE0/0/1端口注册到（加入进）动态VLAN 10中。然后，S2会通过其GE0/0/2端口向外发送VLAN属性注册报文。需要注意的是，只有从链路上接收到VLAN属性注册报文的端口才会注册到相应的VLAN中，所以S2的GE0/0/2端口现在并未注册到动态VLAN 10中。

S3通过其GE0/0/1端口接收到S2发送过来的VLAN属性注册报文后，会自动创建动态VLAN 10，并将自己的GE0/0/1端口注册到动态VLAN 10中。然后，S3会通过其GE0/0/2端口向外发送VLAN属性注册报文。需要注意的是，S3的GE0/0/2端口现在并未注册到动态VLAN 10中。

S4通过其GE0/0/1端口接收到S3发送过来的VLAN属性注册报文后，会自动创建动态VLAN 10，并将自己的GE0/0/1端口注册到动态VLAN 10中。

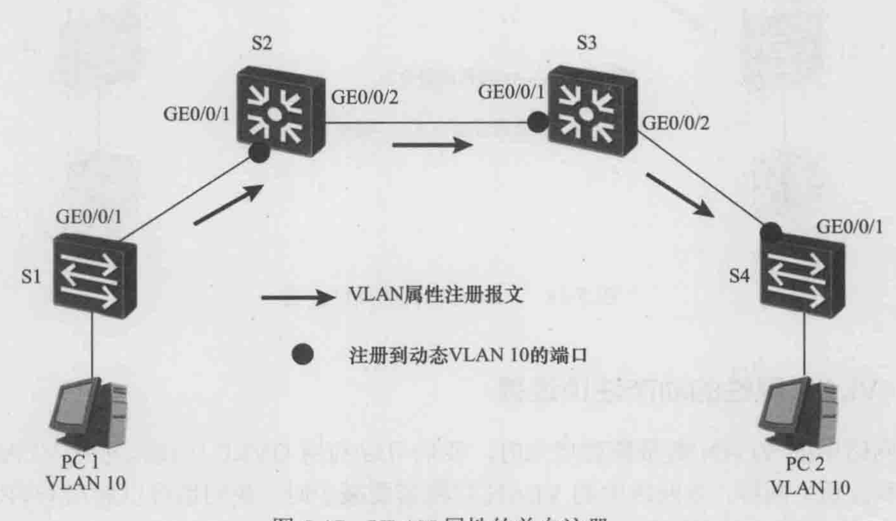


图5-17　VLAN属性的单向注册

在上述由 S1 向 S4 传递关于 VLAN 10 的信息过程中，S2、S3、S4 上自动创建了动态 VLAN 10，并且 S2 的 GE0/0/1 端口、S3 的 GE0/0/1 端口、S4 的 GE0/0/1 端口已经注册到 VLAN 10 中（这个过程称为 VLAN 属性的单向注册过程）。但是，到目前为止，S2 的 GE0/0/2 端口和 S3 的 GE0/0/2 端口还未注册到 VLAN 10 中，所以，用户还必须在 S4 上手工创建静态 VLAN 10，并配置 S4 的 GE0/0/1 端口允许通过属于 VLAN 10 的帧。

如图 5-18 所示，在 S4 上手工创建静态 VLAN 10，并配置 S4 的 GE0/0/1 端口允许通过属于 VLAN 10 的帧。注意，所配置的静态 VLAN 10 的信息会替换掉 S4 上的动态 VLAN 10 的信息。然后，S4 的 GE0/0/1 端口会向外发送 VLAN 属性的注册报文。

S3 通过其 GE0/0/2 端口接收到 S4 发送过来的 VLAN 属性注册报文后，会将自己的 GE0/0/2 端口注册到已经被创建了的动态 VLAN 10 中。然后，S3 会通过其 GE0/0/1 端口向外发送 VLAN 属性注册报文。

S2 通过其 GE0/0/2 端口接收到 S3 发送过来的 VLAN 属性注册报文后，会将自己的 GE0/0/2 端口注册到已经被创建了的动态 VLAN 10 中。然后，S2 会通过其 GE0/0/1 端口向外发送 VLAN 属性注册报文。

S1 的 GE0/0/1 端口也会接收到 S2 发送过来的 VLAN 属性注册报文。由于 S1 上以及 S1 的 GE0/0/1 端口上已经有了关于静态 VLAN 10 的相关信息，所以 S1 不会进行涉及动态 VLAN 10 的相关操作。

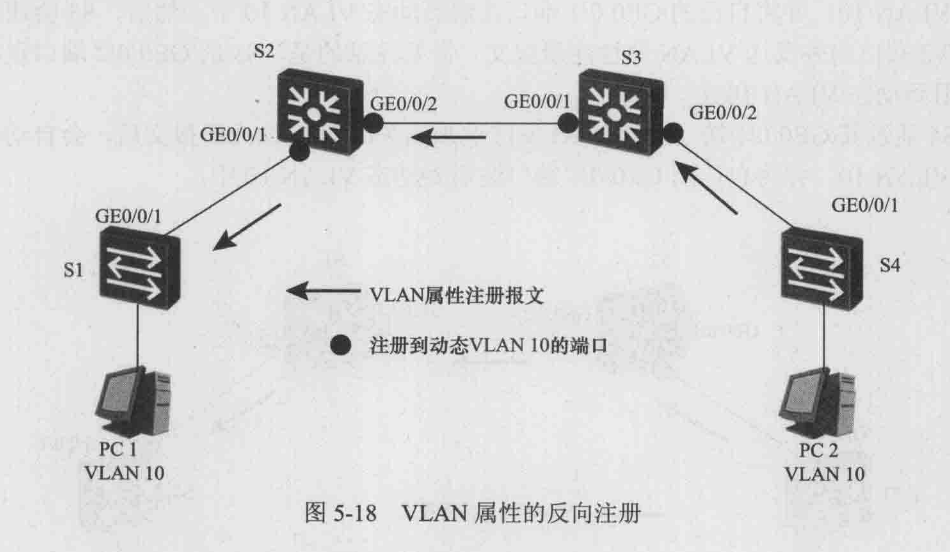


图 5-18　VLAN 属性的反向注册

5.8.2 VLAN 属性的动态注销过程

当网络中的 VLAN 数量需要增加时，我们可以利用 GVRP 协议来进行 VLAN 的自动创建和注册。同样，当网络中的 VLAN 数量需要减少时，我们也可以利用 GVRP 协议来进行 VLAN 的自动删除和注销。

如图 5-19 所示，当 PC 1 和 PC 2 不再属于 VLAN 10 时，用户就需要在 S1 上

手工删除静态 VLAN 10。然后，S1 的 GE0/0/1 端口会向外发送 VLAN 属性的注销报文。

S2 通过其 GE0/0/1 端口接收到 S1 发送过来的 VLAN 属性注销报文后，会将自己的 GE0/0/1 端口从动态 VLAN 10 中注销。然后，S2 会通过其 GE0/0/2 端口向外发送 VLAN 属性注销报文。需要注意的是，只有从链路上接收到 VLAN 属性注销报文的端口才会从相应的动态 VLAN 中被注销，所以 S2 的 GE0/0/2 端口现在并未从动态 VLAN 10 中被注销。另外，由于此时 S2 上的动态 VLAN 10 中还存在 GE0/0/2 端口，所以 S2 上的动态 VLAN 10 还会继续存在。

S3 通过其 GE0/0/1 端口接收到 S2 发送过来的 VLAN 属性注销报文后，会将自己的 GE0/0/1 端口从动态 VLAN 10 中注销。然后，S3 会通过其 GE0/0/2 端口向外发送 VLAN 属性的注销报文。由于此时 GE0/0/2 端口并未从动态 VLAN 10 中被注销，所以 S3 上的动态 VLAN 10 还会继续存在。

S4 通过其 GE0/0/1 端口接收到 S3 发送过来的 VLAN 属性注销报文后，不会进行涉及动态 VLAN 10 的相关操作，原因是 S4 上的 VLAN 10 并非动态 VLAN，而是手工创建的静态 VLAN。

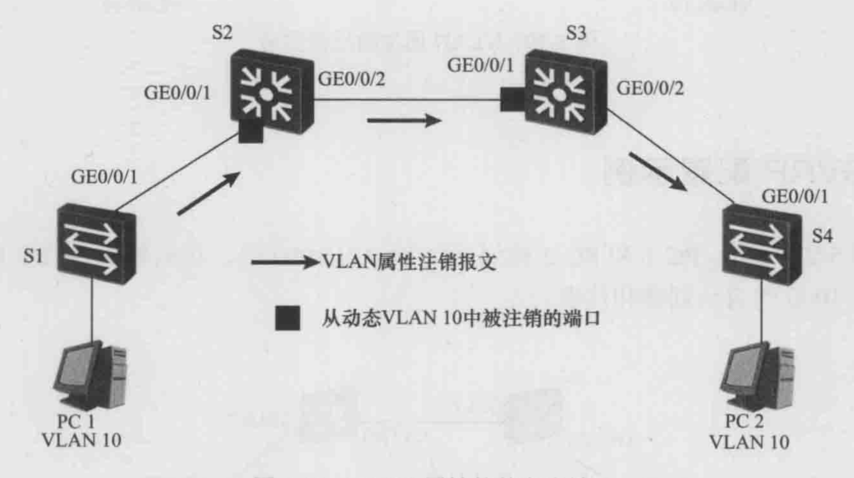


图 5-19 VLAN 属性的单向注销

通过上述 VLAN 属性的注销过程可以看到，S2 的 GE0/0/1 端口和 S3 的 GE0/0/1 端口已经从动态 VLAN 10 中被注销（这个过程称为 VLAN 属性的单向注销过程），但是，S2 和 S3 上还仍然存在动态 VLAN 10，S2 的 GE0/0/2 端口和 S3 的 GE0/0/2 端口尚未在动态 VLAN 10 中被注销。为此，用户还必须在 S4 上手工删除静态 VLAN 10。如图 5-20 所示，当 S4 上的静态 VLAN 10 被手工删除之后，S4 的 GE0/0/1 端口便会向外发送 VLAN 属性的注销报文。

S3 通过其 GE0/0/2 端口接收到 S4 发送过来的 VLAN 属性注销报文后，会将自己的 GE0/0/2 端口从动态 VLAN 10 中注销。然后，S3 会通过其 GE0/0/1 端口向外发送 VLAN 属性的注销报文。由于现在已经没有任何 S3 的端口注册在动态 VLAN 10 中，所以 S3 上的动态 VLAN 10 会被自动删除掉。

S2 通过其 GE0/0/2 端口接收到 S3 发送过来的 VLAN 属性注销报文后，会将自己的 GE0/0/2 端口从动态 VLAN 10 中注销。然后，S2 会通过其 GE0/0/1 端口向外发送 VLAN 属性的注销报文。由于现在已经没有任何 S2 的端口注册在动态 VLAN 10 中，所以 S2 上的动态 VLAN 10 会被自动删除掉。

S1 通过其 GE0/0/1 端口接收到 S2 发送过来的 VLAN 属性注销报文后，不会进行涉及动态 VLAN 10 的相关操作，原因是 S4 上已经不存在 VLAN 10。



图 5-20 VLAN 属性的反向注销

5.9 GVRP 配置示例

如图 5-21 所示，PC 1 和 PC 2 被划分至 VLAN 1000 中，我们希望 GVRP 协议来实现 VLAN 1000 的自动创建和注册。

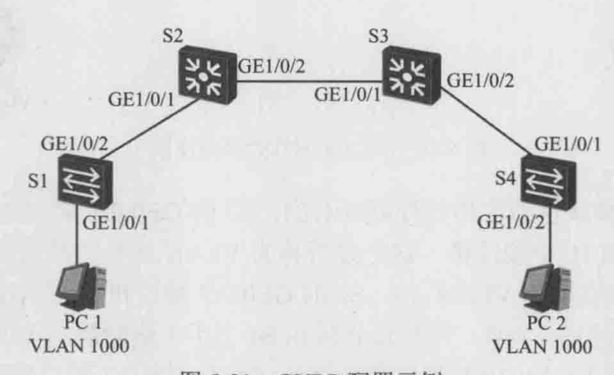


图 5-21 GVRP 配置示例

1. 配置思路

（1）在每台交换机的全局及端口下使能 GVRP 功能。

（2）配置交换机的二层连通性，即将交换机的某些端口配置为 Trunk 端口并允许相应的 VLAN 帧通过。

（3）在交换机 S1 和 S4 上配置静态 VLAN 1 000。

2. 配置步骤

在交换机的系统视图下执行命令 **gvrp** 来全局使能 GVRP 功能。

#配置 S1，在 S1 上全局使能 GVRP 功能。

```
<Quidway> system-view
[Quidway] sysname S1
[S1] gvrp
```

#配置 S2，在 S2 上全局使能 GVRP 功能。

```
<Quidway> system-view
[Quidway] sysname S2
[S2] gvrp
```

#配置 S3，在 S3 上全局使能 GVRP 功能。

```
<Quidway> system-view
[Quidway] sysname S3
[S3] gvrp
```

#配置 S4，在 S4 上全局使能 GVRP 功能。

```
<Quidway> system-view
[Quidway] sysname S4
[S4] gvrp
```

全局使能 GVRP 之后，还需要对相关的端口进行配置。

（1）使能端口的 GVRP 功能。注意，在使能端口的 GVRP 功能之前，需要先全局使能 GVRP 功能。

（2）配置相关的端口为 Trunk 端口，并允许相应的 VLAN 通过。注意，GVRP 功能只能配置在 Trunk 类型的端口上。

#配置 S1 的端口。

```
[S1] interface gigabitethernet 1/0/1
[S1-GigabitEthernet1/0/1] port link-type access
[S1-GigabitEthernet1/0/1] port default vlan 1000
[S1-GigabitEthernet1/0/1] quit
[S1] interface gigabitethernet 1/0/2
[S1-GigabitEthernet1/0/2] gvrp
[S1-GigabitEthernet1/0/2] port link-type trunk
[S1-GigabitEthernet1/0/2] port trunk allow-pass vlan all
[S1-GigabitEthernet1/0/2] quit
```

#配置 S2 的端口。

```
[S2] interface gigabitethernet 1/0/1
[S2-GigabitEthernet1/0/1] gvrp
[S2-GigabitEthernet1/0/1] port link-type trunk
[S2-GigabitEthernet1/0/1] port trunk allow-pass vlan all
[S2-GigabitEthernet1/0/1] quit
[S2] interface gigabitethernet 1/0/2
[S2-GigabitEthernet1/0/2] gvrp
[S2-GigabitEthernet1/0/2] port link-type trunk
[S2-GigabitEthernet1/0/2] port trunk allow-pass vlan all
[S2-GigabitEthernet1/0/2] quit
```

#配置 S3 的端口。

```
[S3] interface gigabitethernet 1/0/1
[S3-GigabitEthernet1/0/1] gvrp
[S3-GigabitEthernet1/0/1] port link-type trunk
```

```
[S3-GigabitEthernet1/0/1] port trunk allow-pass vlan all
[S3-GigabitEthernet1/0/1] quit
[S3] interface gigabitethernet 1/0/2
[S3-GigabitEthernet1/0/2] gvrp
[S3-GigabitEthernet1/0/2] port link-type trunk
[S3-GigabitEthernet1/0/2] port trunk allow-pass vlan all
[S3-GigabitEthernet1/0/2] quit
```

#配置 S4 的端口。

```
[S4] interface gigabitethernet 1/0/1
[S4-GigabitEthernet1/0/1] gvrp
[S4-GigabitEthernet1/0/1] port link-type trunk
[S4-GigabitEthernet1/0/1] port trunk allow-pass vlan all
[S4-GigabitEthernet1/0/1] quit
[S4] interface gigabitethernet 1/0/2
[S4-GigabitEthernet1/0/2] port link-type access
[S4-GigabitEthernet1/0/2] port default vlan 1000
[S4-GigabitEthernet1/0/2] quit
```

接下来，我们在 S1 和 S4 上手动创建静态 VLAN 1000，那么 GVRP 就会自动完成 S2 和 S3 上的 VLAN 配置。配置之前先查看一下 S2 和 S3 上的 VLAN 信息，在完成配置之后我们再来查看一下 S2 和 S3 上的 VLAN 信息。通过对比，就能看出 GVRP 的作用。

#以 S2 为例，在系统视图下执行命令 **display vlan summary**，查看 VLAN 信息。

```
[S2] display vlan summary
static vlan:
Total 1 static vlan.
  1

dynamic vlan:
Total 0 dynamic vlan.
```

可以看到，S2 现在上没有任何动态 VLAN 的信息。接下来，在 S1 和 S4 上手动创建静态 VLAN 1000。

#配置 S1，在 S1 上创建静态 VLAN 1000。

```
[S1] vlan 1000
[S1-VLAN1000] quit
```

#配置 S4，在 S4 上创建静态 VLAN 1000。

```
[S4] vlan 1000
[S4-VLAN1000] quit
```

在完成上述配置后，我们可以通过以下命令来对配置进行验证。

1）使用命令 **display gvrp status**，查看全局 GVRP 的使能情况。

2）使用命令 **display gvrp statistics**，查看端口的 GVRP 统计信息。

#在 S1 上使用命令 **display gvrp status**，查看全局 GVRP 的使能情况。

```
[S1] display gvrp status
 Info: GVRP is enabled
```

可以看到，S1 上已经全局使能了 GVRP 功能。

#在 S2 上使用命令 **display gvrp statistics**，查看端口的 GVRP 统计信息。

```
[S2] display gvrp statistics
 GVRP statistics on port GigabitEthernet1/0/1
 GVRP status                              : Enabled
```

```
GVRP registrations failed          : 0
GVRP last PDU origin               : 0000-0000-0000
GVRP registration type             : Normal

GVRP statistics on port GigabitEthernet1/0/2
GVRP status                        : Enabled
GVRP registrations failed          : 0
GVRP last PDU origin               : 0000-0000-0000
GVRP registration type             : Normal
```

回显信息中的“GVRP status：Enabled”，说明 S2 的 GE1/0/1 端口和 GE1/0/2 端口已经使能了 GVRP 功能。

最后，让我们来看看 S2 上 VLAN 的信息发生了哪些变化。

#在 S2 上使用命令 **display vlan summary**，查看 VLAN 信息。

```
[S2] display vlan summary
static vlan:
Total 1 static vlan.
  1

dynamic vlan:
Total 1 dynamic vlan.
  1000
```

从回显信息中可以看到，S2 上已经存在动态 VLAN 1000。

5.10 练习题

1. （多选）下列关于 VLAN 的描述中，错误的是（ ）
   A．VLAN 技术可以将一个规模较大的冲突域隔离成若干个规模较小的冲突域
   B．VLAN 技术可以将一个规模较大的二层广播域隔离成若干个规模较小的二层广播域
   C．位于不同 VLAN 中的计算机之间无法进行通信
   D．位于同一 VLAN 中的计算机之间可以进行二层通信
2. （多选）下列关于 VLAN 的描述中，正确的是（ ）
   A．IEEE 802.1Q 帧的 Tag 中的 VID 的实际有效值可以是 1
   B．IEEE 802.1Q 帧的 Tag 中的 VID 的实际有效值可以是 1 024
   C．IEEE 802.1Q 帧的 Tag 中的 VID 的实际有效值可以是 2 048
   D．IEEE 802.1Q 帧的 Tag 中的 VID 的实际有效值可以是 4 096
3. （多选）下列关于 VLAN 的描述中，错误的是（ ）
   A．计算机可以产生并发送带 Tag 的 IEEE 802.1Q 帧
   B．部署 VLAN 时，交换机和计算机上都需要进行 VLAN 相关的配置
   C．在现实网络中，应用最为广泛的 VLAN 是三层 VLAN
4. （多选）下列关于 VLAN 的描述中，正确的是（ ）
   A．交换机上直连计算机的端口通常应配置为 Access 端口
   B．交换机上直连计算机的端口通常应配置为 Trunk 端口

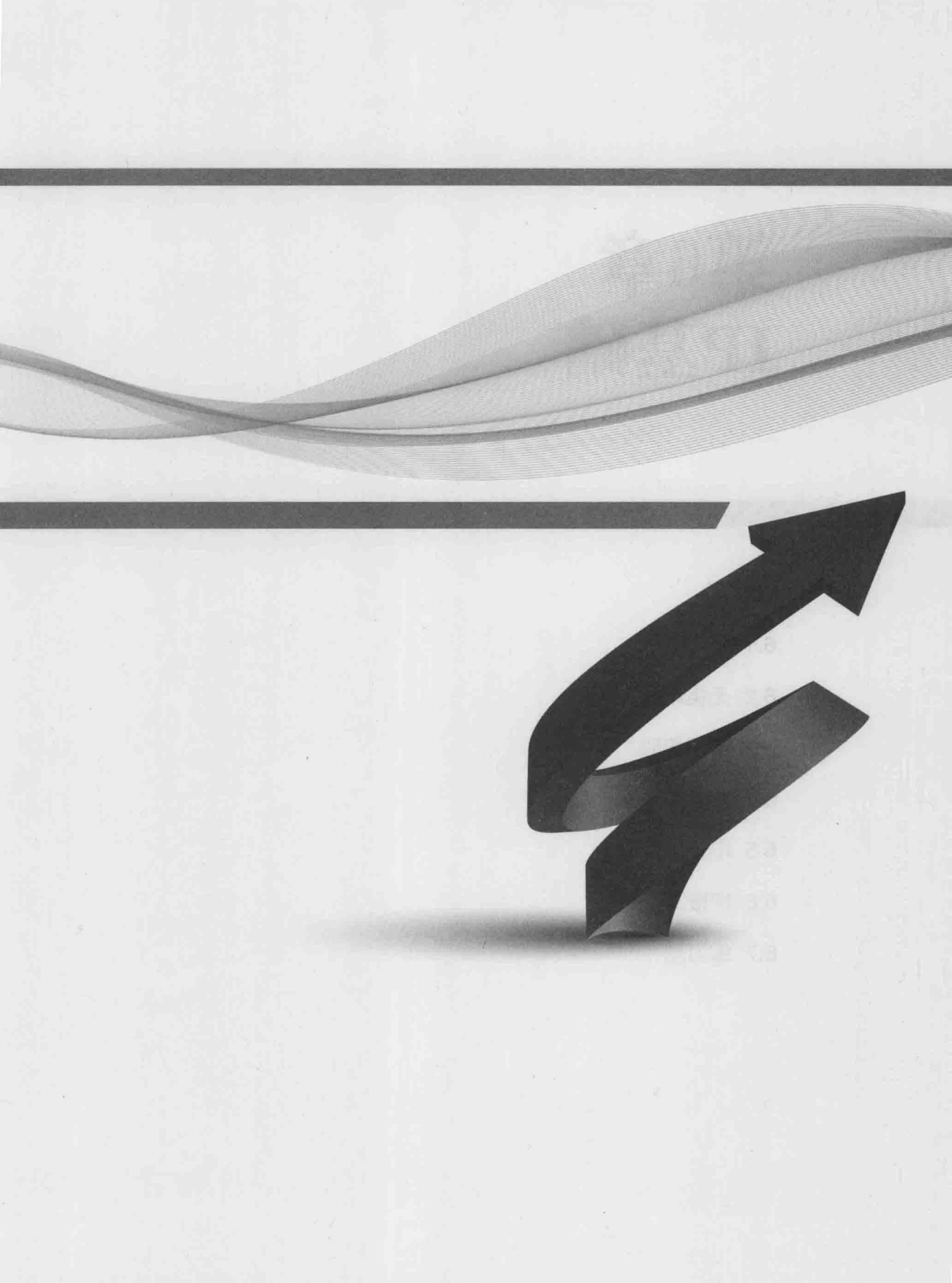
C．在 Access 链路上运动的帧应该是带 Tag 的

5．（多选）下列关于 GVRP 的描述中，正确的是（ ）

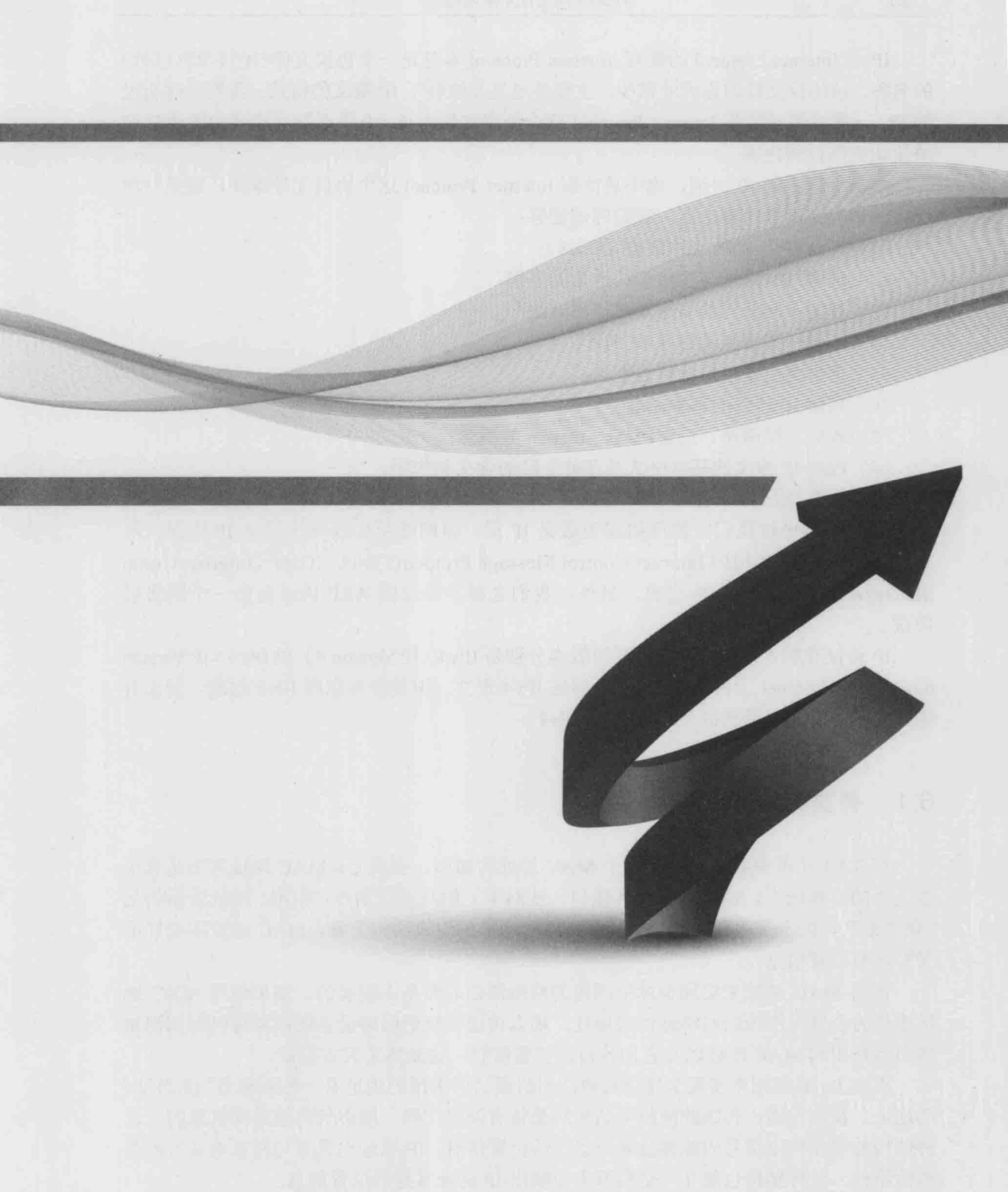
A．GVRP 是 Generic VLAN Registration Protocol 的简称

B．GVRP 是 GARP VLAN Registration Protocol 的简称

C．GVRP 可用减少手工配置 VLAN 的工作量



第6章 IP基础



IP 是 Internet Protocol 的缩写。Internet Protocol 本身是一个协议文件（IETF RFC 791）的名称，该协议文件的内容非常少，主要是定义并阐释了 IP 报文的格式。我们平时所说的 IP，一般并不是特指 Internet Protocol 这个协议文件本身，而是泛指直接或间接地与 IP 协议相关的任何内容。

本章标题中的 IP 一词，也不是特指 Internet Protocol 这个协议文件本身，而是一种泛指。学习完本章内容之后，我们应该能够：

（1）理解 IP 地址的 5 种分类（Class）；

（2）理解网络地址与主机接口地址的区别；

（3）理解子网掩码的作用及其使用方法；

（4）了解一些常用的特殊 IP 地址；

（5）理解路由器接口的行为特点；

（6）理解 IP 转发的基本原理；

（7）理解二层网络、三层网络、internet 等概念；

（8）理解 IP 报文的基本格式及部分字段的含义和作用。

从图 1-7 我们可以看出，IP 协议是 TCP/IP 协议簇中网络层的一个协议。需要说明的是，虽然平时我们常把网络层说成是 IP 层，但网络层协议并不只是 IP 协议，网络层协议还包括 ICMP（Internet Control Message Protocol）协议、IGMP（Internet Group Management Protocol）协议等。另外，我们之前学习过的 ARP 协议也是一个网络层协议。

IP 协议有版本之分。两个重要的版本分别是 IPv4（IP Version 4）和 IPv6（IP Version 6）。目前，Internet 上的 IP 报文主要都是 IPv4 报文，但是逐步在向 IPv6 过渡。若无特别声明，本章及以后所提及的 IP 均指 IPv4。

6.1 有类编址

在 3.2.1 小节中我们学习了关于 MAC 地址的知识。实质上，MAC 地址并不是真正意义上的“地址”，而是某个设备接口（或网卡）的身份识别号：MAC 地址表示的是“我是谁”，而不是“我在哪里”。从 MAC 地址的组成结构上看，MAC 地址本身并不带有任何位置信息。

使用 MAC 地址来实现全球范围内的网络通信显然是不现实的。如果使用 MAC 地址来作为全球范围内的网络通信的地址，那么传递信息的网络设备就需要每时每刻都知道所有在用的 MAC 地址以及它们各自的位置信息，这显然是天方夜谭。

事实上，真正用来实现全球范围内的网络通信所采用的地址是一种被称为“IP 地址”的地址。我们知道，传统的座机电话号码是带有国家代码、城市代码等结构信息的，这种结构使得座机电话号码能够反映出自己的位置信息。IP 地址也具有与座机电话号码类似的结构，这种结构也能在一定程度上反映出 IP 地址本身的位置信息。

与 MAC 地址一样，IP 地址是网络设备接口的属性，而不是网络设备本身的属性。当我们说给某台设备分配一个 IP 地址时，实质上是指给这台设备的某个接口分配一个 IP

地址；设备有多个接口时，通常每个接口都至少需要一个 IP 地址。

需要使用 IP 地址的接口通常是路由器和计算机的接口。交换机的接口（端口）通常是不需要 IP 地址的（注意，这里所说的交换机是指不具备网络层转发功能的“二层交换机”）。在谈及 IP 地址分配的问题时，常常把路由器和计算机统称为“主机（Host）”，并且常常把主机的某个（或某些）接口的 IP 地址简称为主机 IP 地址。

IP 地址的长度是 32 个比特，由 4 个字节组成。为了阅读和书写的方便，IP 地址通常采用点分十进制数来表示。例如，11.1.0.254 就是一个采用点分十进制数表示的 IP 地址，表 6-1 给出了它所对应的二进制数。

表 6-1　IP 地址的二进制格式与十进制格式对比

进制	第一字节	第二字节	第三字节	第四字节
十进制	11	1	0	254
二进制	00001011	00000001	00000000	11111110

IP 地址是统一由 ICANN（Internet Corporation for Assigned Names and Numbers）来分配和管理的。IP 地址的分配有一套严格的机制和程序，这种机制和程序保证了 IP 地址在 Internet 上的唯一性。

IP 地址最初被设计划分成了 5 类（Class），分别称为 A 类、B 类、C 类、D 类、E 类，如图 6-1 所示。

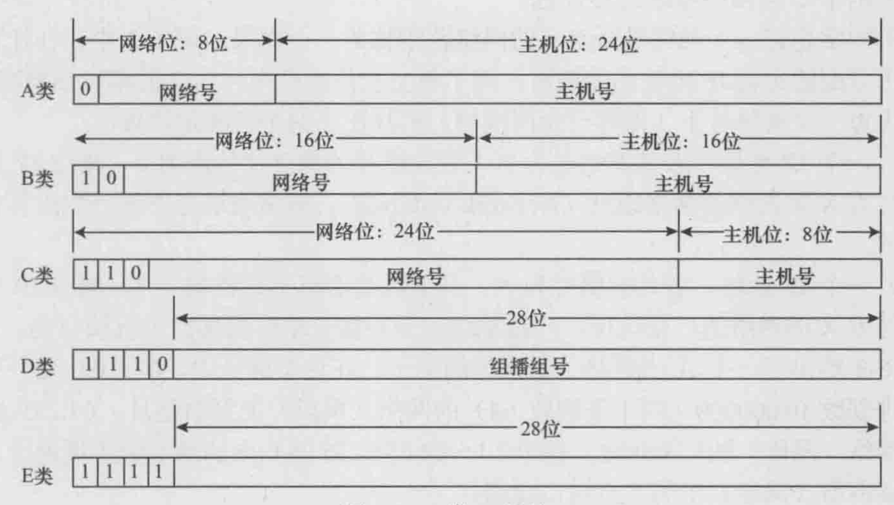


图 6-1　5 类 IP 地址

在这 5 类 IP 地址中，D 类地址属于组播 IP 地址的范畴（注意，不要与组播 MAC 地址混淆了，虽然二者具有一定的相似性），E 类地址是专门用于特殊的实验目的的。我们这里只关注 A、B、C 三类地址。

A、B、C 三类地址都是单播 IP 地址（其中的一些特殊地址除外），只有这三类中的地址才能分配给主机接口使用。主机接口的 IP 地址既是该接口在网络层的“身份识别号”，又在一定程度上表示了该接口的位置信息。

从图 6-1 可以看出，主机 IP 地址分为网络号（Netid）和主机号（Hostid）两部分。

网络号用于表示主机接口所在的网络，类似于“××省××市××区××街道”的作用，而主机号用于表示在网络号所定义的网络范围内某个特定的主机接口，类似于“门牌号”的作用。

我们把使用 A 类地址的网络称为 A 类网络，使用 B 类地址的网络称为 B 类网络，使用 C 类地址的网络称为 C 类网络。从图 6-1 可以看出，A 类网络的网络号的个数很少，但每个 A 类网络中所允许的主机接口的个数却非常多；相反，C 类网络的网络号的个数非常多，但每个 C 类网络中所允许的主机接口的个数却非常少；B 类网络的情况介于二者之间，具体的数量关系如表 6-2 所示。

表 6-2 三类地址的结构差异

	网络号位数	网络号个数	主机号位数	每个网络号下可分配的主机 IP 地址的个数	地址范围
A 类	8	2^7=128	24	2^{24}-2=16 777 214	0.0.0.0～127.255.255.255
B 类	16	2^{14}=16 384	16	2^{16}-2=65 534	128.0.0.0～191.255.255.255
C 类	24	2^{21}=2 097 152	8	2^8-2=254	192.0.0.0～223.255.255.255

网络号与主机号这种二分结构，使得 IP 地址的分配在一定程度上具有了合理性和灵活性。比如，对于一个大型机构的网络（假设该网络包含了 1.6×10^7 个主机接口），则给它分配一个 A 类网络号就比较合适；而对于一个只包含 500 个主机接口的小型网络，则给它分配两个 C 类网络号就比较合适。

我们通常也把一个网络号所定义的网络范围称为一个网段。表 6-2 中，在计算一个网段中可分配的主机 IP 地址的个数时，除了将主机号的位数作为 2 的幂，还要减去 2，这是因为每一个网络号下（即每一个网段中）都预留了两个特殊的地址。

（1）一个 IP 地址，若其网络号为 X，且主机号的每个比特均为 0，则该 IP 地址称为网络号为 X 的网络的网络地址（Network Address）。网络地址是不能分配给具体的主机接口的。

（2）一个 IP 地址，若其网络号为 X，且主机号的每个比特均为 1，则该 IP 地址称为网络号为 X 的网络的广播地址。广播地址也是不能分配给具体的主机接口的。

表 6-3 给出了一个 A 类网络（网段）的例子。在这个例子中，64.0.0.0 是一个网络号为二进制数 01000000（或十进制数 64）的网络（网段）的网络地址；64.255.255.255 是这个网络（网段）的广播地址；64.0.0.1～64.255.255.254 中的地址为主机地址，可以分配给该网络（网段）中的主机接口使用。

表 6-3 一个 A 类网段示例

网络号		主机号	点分十进制格式	类型
固定位	其他位			
0	1000000	00000000 00000000 00000000	64.0.0.0	网络地址
		00000000 00000000 00000001～11111111 11111111 11111110	64.0.0.1～64.255.255.254	主机地址
		11111111 11111111 11111111	64.255.255.255	广播地址

在网络通信发展的初期，网络中的计算机数量很少，需要使用的 IP 地址也很少。

所以，这种将 IP 地址划分为 5 类的做法在当时看来并没有什么问题。然而，随着网络通信的迅猛发展，这种称为“有类编址（Classful Addressing）”的地址划分方法却暴露出了明显的问题。例如，某个大公司需要建设一个规模较大的网络，需要大约十万个 IP 地址，假设 B 类网络号早已被分配完毕，那么如果给这个网络分配一个 A 类网络号，则意味着将有大约 1.66×10^7 个 IP 地址被浪费掉。类似的例子数不胜数。总之，“有类编址”的地址划分方法太过于死板，也可以说是划分的颗粒度太大，使得拥有大量主机号的 A 类和 B 类地址不能被充分地利用起来，从而造成了大量的 IP 地址资源浪费。

6.2 无类编址

有类编址方法中，A 类、B 类、C 类地址限定了网络号和主机号的位数。无类编址（Classless Addressing）则是不限定网络号和主机号的位数，这使得 IP 地址的分配更加灵活，IP 地址的利用率也得到了提高。

比如，原来的 64.0.0.0 这个 A 类网段内的 IP 地址如果都分配给一个组织，则无法利用的 IP 地址就会太多太多。现在，我们可以扩展网络号的位数，减少主机号的位数，就可以使得这个范围内的 IP 地址可以分配给更多的组织，同时减少 IP 地址的浪费。假设我们希望将这个范围内的地址分成 4 部分，分别分配给 4 个组织，则可以按表 6-4 所示的方案进行分配。

表 6-4 扩展网络号的位数

	网络号	主机号的位数	可分配的 IP 地址个数
有类编址	0100 0000	24	$2^{24}-2$=16 777 214
无类编址	0100 0000 00	22	$2^{22}-2$=4 194 302
	0100 0000 01	22	$2^{22}-2$=4 194 302
	0100 0000 10	22	$2^{22}-2$=4 194 302
	0100 0000 11	22	$2^{22}-2$=4 194 302

可以看到，保持原来的网络位不变，从以前的主机号中拿出前两位用于网络位，就可以将原来的一整段 IP 地址分成 4 个新的网段。每个新网段内所包含的 IP 地址的数量都有所减少，但这些 IP 地址却是可以分配给 4 个不同的组织。

通常，我们可以这样来规划和分配 IP 地址：假设一个组织所需的主机 IP 地址的数量为 N，我们可以通过计算确定出大于或等于 N+2 的最小的 2 的幂，然后以幂的值作为主机号的位数，余下的位全部作为网络位。

例如，公司 X 申请到了一个网络地址为 192.168.1.0 的 C 类网段，这个网段原来的网络位是 24bit，主机位是 8bit，共有 254 个地址可供分配（192.168.1.1～192.168.1.254）。但是，X 公司有 3 个独立的部门 X1、X2、X3，每个部门都需要建立自己的网络，并且要求不同部门的网络所使用的网络号不能相同。这 3 个部门的网络所需要的主机 IP 地址个数分别是 100、50、30。那么，我们可以根据表 2-5 所示的方案来合理地将 IP 地址分配给这 3 个部门的网络。

表 6-5 使用无类编址进行 IP 地址的分配

部门	所需地址数	大于或等于 N+2 的最小的 2 的幂	主机号位数	网络号位数	网络位（省略固定的前 24 位）	主机位范围	地址范围	可分配的地址个数
X1	100	$128=2^7$	7	32–7=25	0	0000000～1111111	192.168.1.0～192.168.1.127	$2^7-2=126$
X2	50	$64=2^6$	6	32–6=26	10	000000～111111	192.168.1.128～192.168.1.191	$2^6-2=62$
X3	30	$32=2^5$	5	32–5=27	110	00000～11111	192.168.1.192～192.168.1.223	$2^5-2=30$
剩余	—	—	5	27	111	00000～11111	192.168.1.224～192.168.1. 255	$2^5-2=30$

采用有类编址方式时，我们很容易知道关于一个 IP 地址的所有信息。例如，对于 64.1.5.0 这个 IP 地址，由于其第一个字节的值在 0～127 的范围内，所以它肯定是一个 A 类地址，于是 64 便是其所在网络的网络号，其余 3 个字节为其主机号。并且，64.0.0.0 是这个网络的网络地址，64.255.255.255 是这个网络的广播地址，64.1.5.0 是这个网络中的一个主机接口地址。

然而，采用无类编址方式后，情况就不一样了。同样是 64.1.5.0 这个地址，它可能是网络号为前两个字节（64.1）的网络中的一个主机接口地址，也可能是网络号为前 3 个字节（64.1.5）的网络的网络地址，并且还有很多的其他可能性。

那么，采用无类编址方式时，我们如何才能判断出一个 IP 地址所属网络的网络号呢？要回答这个问题，就必须引入子网掩码的概念。

6.3 子网掩码

子网掩码（Subnet Mask）由 32 个比特组成，也可看作是由 4 个字节组成，并且也通常以点分十进制数来表示。但是，子网掩码本身并不是一个 IP 地址，并且子网掩码必须由若干个连续的 1 后接若干个连续的 0 组成。下面是一些例子。

11111100 00000000 00000000 00000000（252.0.0.0）子网掩码

11111111 11000000 00000000 00000000（255.192.0.0）子网掩码

11111111 11111111 11111111 11110000（255.255.255.240）子网掩码

11111111 11111111 11111111 11111111（255.255.255.255）子网掩码

00000000 00000000 00000000 00000000（0.0.0.0）子网掩码

11011000 00000000 00000000 00000000（216.0.0.0）不是子网掩码

00000000 11111111 11111111 11111111（0.255.255.255）不是子网掩码

我们通常将一个子网掩码中 1 的个数称为这个子网掩码的长度。例如，子网掩码 0.0.0.0 的长度为 0，子网掩码 252.0.0.0 的长度为 6，子网掩码 255.192.0.0 的长度为 10，子网掩码 255.255.255.255 的长度为 32。

子网掩码总是与 IP 地址结合使用的。当一个子网掩码与一个 IP 地址结合使用时，

子网掩码中1的个数（也就是子网掩码的长度）就表示这个IP地址的网络号的位数，而0的个数就表示这个IP地址的主机号的位数。如果将一个子网掩码与一个IP地址进行逐位的“与”运算，所得的结果便是该IP地址所在网络的网络地址。

例如，对于64.1.5.0这个IP地址，假设其子网掩码为255.255.0.0，那么我们就可以通过计算得知这个IP地址所在网络的网络地址为64.1.0.0，计算过程如表6-6所示。

表6-6　从IP地址和子网掩码到网络地址

	第一字节	第二字节	第三字节	第四字节
IP地址	01000000	00000001	00000101	00000000
子网掩码	11111111	11111111	00000000	00000000
逐位“与”运算结果	01000000	00000001	00000000	00000000
网络地址	64	1	0	0

子网掩码的引入，使得无类编址方式可以完全后向兼容有类编址方式，即：有类编址时，A类地址的子网掩码总是255.0.0.0，B类地址的子网掩码总是255.255.0.0，C类地址的子网掩码总是255.255.255.0。这样一来，所谓的有类编址便成为了无类编址的特例。使用无类编址时，子网掩码的长度是可以根据需要而灵活变化的，所以此时的子网掩码也称为“可变长子网掩码（Variable Length Subnet Mask，VLSM）”。

目前，Internet所使用的编址方式都是无类编址方式，一个IP地址总是有其对应的子网掩码。我们在书写IP地址及其对应的子网掩码时，习惯IP地址在前，子网掩码在后，中间以“/”隔开；另外，为了简化起见，还常常以子网掩码的长度来代替子网掩码本身。例如：64.1.5.0/255.255.0.0（或64.1.5.0/16）、192.168.1.5/252.0.0.0（或192.168.1.5/6）。

6.4　特殊IP地址

我们曾提到IP地址是由ICANN来统一分配的，以保证任何一个IP地址在Internet上的唯一性。其实，这里的IP地址是指公网IP地址。连接到Internet的网络设备必须具有由ICANN分配的公网IP地址。

但是，实际上有一些网络并不需要连接到Internet，比如一个大学的封闭实验室内的网络。这种网络中的网络设备无须使用公网IP地址，只要同一网络中的网络设备的IP地址不发生冲突即可。

在IP地址的空间里，A、B、C三类地址中各预留了一些地址专门用于上述情况，它们被称为私网IP地址，如下。

（1）A类：10.0.0.0～10.255.255.255。

（2）B类：172.16.0.0～172.31.255.255。

（3）C类：192.168.0.0～192.168.255.255。

凡是Internet上的网络设备均不会接收、发送或者转发源IP地址或目的IP地址在上述范围内的报文，这些IP地址只能用于私有网络。私网地址的使用使得网络可以得到更为自由的扩展，因为同一个私网IP地址是可以在不同的私有网络中得到重

复使用的。

本来，私有网络由于使用了私网 IP 地址，所以是不允许连接到 Internet 上的。然而，由于实际需求的驱动，许多私有网络也希望能够连接到 Internet 上，从而实现私网与 Internet 之间的通信，以及通过 Internet 实现私网与私网之间的通信，如图 6-2 所示。私网与 Internet 的互联，必须使用到一种被称为“网络地址转换（Network Address Translation，NAT）”的技术。关于 NAT 技术，后面的章节会有专门的介绍。

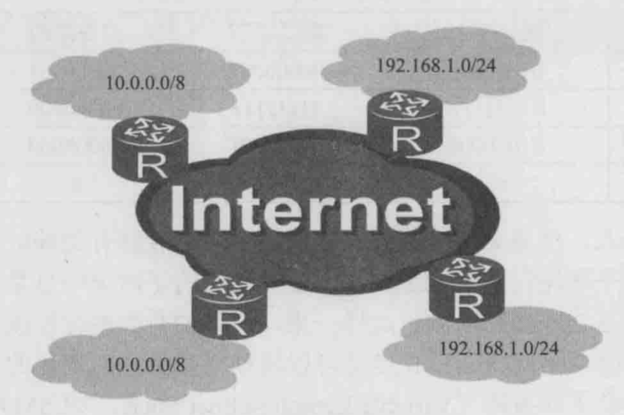


图 6-2　私有网络连接到 Internet 上

IP 地址空间中，除了私网 IP 地址外，还有不少其他的特殊 IP 地址，这些 IP 地址有着特殊的含义和作用，举例如下。

255.255.255.255

这个地址称为有限广播地址（Limited Broadcast Address），它可以作为一个 IP Packet 的目的 IP 地址使用。路由器接收到目的 IP 地址为有限广播地址的 IP Packet 后，会停止对该 IP Packet 的转发。

0.0.0.0

如果把这个地址作为一个网络地址来对待，它的意思便是“任何网络”的网络地址。如果把这个地址作为一个主机接口地址来对待，它的意思便是“这个网络上这个主机接口”的 IP 地址。例如，当一个主机接口在启动过程中尚未获得自己的 IP 地址时，就可以向网络发送目的 IP 地址为有限广播地址、源 IP 地址为 0.0.0.0 的 DHCP（Dynamic Host Configuration Protocol）请求报文，希望 DHCP 服务器在收到自己的请求后，能够给自己分配一个可用的 IP 地址。

127.0.0.0/8

这部分地址称为环回地址（Loopback Address）。环回地址可以作为一个 IP Packet 的目的 IP 地址使用。一个设备所产生的、目的 IP 地址为环回地址的 IP Packet 是不可能离开这个设备本身的。环回地址的作用通常是用来测试设备自身的软件系统。

169.254.0.0/16

如果一个网络设备获取 IP 地址的方式被设置成了自动获取方式，但是该设备在网络上又没有找到可用的 DHCP 服务器，那么该设备会使用 169.254.0.0/16 网段中的某个地址来进行临时通信。

6.5　IP 转发原理

路由器（Router）的工作内容主要分为两个方面，一方面是通过运行路由协议（Routing Protocol）来建立并维护自己的路由表（Routing Table），另一方面是根据自己的路由表对 IP 报文（IP Packet，IP 包）进行转发。路由器对 IP 包的转发也称为 IP 转发，或网络层转发，或三层转发，这也是我们本节学习的主要内容。

与交换机一样，一台路由器上也有若干个转发数据的接口（Interface，或 Port），一个接口的行为也是由与该接口对应的网卡控制的。这些网卡的组成结构是与交换机上的网卡或计算机上的网卡的结构完全一样的，同样包含了 CU、OB、IB、LC、LD、TX、RX 这 7 个模块（请仔细复习 3.1.1、3.1.2 小节的内容）。同样，每个接口的网卡都有自己的 MAC 地址，这个 MAC 地址也通常被称为这个接口的 MAC 地址。

路由器上的接口具有如下的行为特点。

（1）当一个单播帧从线路（传输介质）上进入路由器的一个接口后，这个接口会将这个帧的目的 MAC 地址与自己的 MAC 地址进行比较。如果这两个 MAC 地址不相同，则这个接口会将这个帧直接丢弃。如果这两个 MAC 地址相同，则这个接口会将这个帧的载荷数据提取出来，并根据帧的类型字段值将载荷数据上送给路由器的网络层中的相应模块进行后续处理。

（2）当一个广播帧从线路（传输介质）上进入路由器的一个接口后，这个接口会直接将这个帧的载荷数据提取出来，并根据帧的类型字段值将载荷数据上送给路由器的网络层中的相应模块进行后续处理。

（3）当一个组播帧从线路（传输介质）上进入路由器的一个接口后，情况比较复杂，已超出了本书的知识描述范围，所以我们不予考虑。

为了便于简化而清晰地描述 IP 转发原理的核心内容，我们先做出如下的几点假设。

（1）路由器的每个接口都是以太网接口。

（2）从线路上进入路由器的某个接口的帧是一个单播帧，该帧取名为 X。

（3）X 帧的目的 MAC 地址与这个接口的 MAC 地址是相同的。

（4）X 帧的类型字段的值是 0x0800，也就是说 X 帧的载荷数据是一个 IP 包（IP Packet），该 IP 包取名为 P。

（5）P 是一个单播 IP 包，也就是说 P 的目的 IP 地址是一个单播 IP 地址。

接下来，我们先对 IP 转发及其前后的过程进行一个整体性的描述，然后再对 IP 转发原理进行具体而深入的分析。

（1）X 帧从线路上进入路由器的某个接口后，由于 X 帧的目的 MAC 地址与这个接口的 MAC 地址相同，所以该接口会将 P 提取出来。

（2）由于 X 帧的类型字段的值是 0x0800，所以接口会将 P 上送给路由器的网络层中的 IP 转发模块进行处理。

（3）IP 转发模块收到 P 后，会根据 P 的目的 IP 地址查询自己的路由表。查询路由表后，结果会有两种可能：要么将 P 直接丢弃，要么确定出 P 的出接口（即 P 应该从哪个接口离开路由器），以及 P 的下一跳（Next Hop）IP 地址。

（4）IP 转发模块将 P 下发给出接口，同时将 P 的下一跳 IP 地址告诉出接口。

（5）出接口将 P 封装成一个单播帧，该帧取名为 Y。Y 帧的载荷数据就是 P，Y 帧的类型字段的值为 0x0800，Y 帧的源 MAC 地址就是出接口的 MAC 地址，Y 帧的目的 MAC 地址是 P 的下一跳 IP 地址所对应的 MAC 地址。如果路由器能从自己的 ARP 缓存表中查找到 P 的下一跳 IP 地址所对应的 MAC 地址，则直接将这个 MAC 地址作为 Y 帧的目的 MAC 地址；否则，出接口会发出 ARP 请求，以便获知 P 的下一跳 IP 地址所对应的 MAC 地址（请认真复习 3.4 节）。

（6）出接口将 Y 帧发送到线路（传输介质）上去。

以上 6 个步骤便是 IP 转发的整体过程，如图 6-3 所示。显然，在这 6 个步骤中，第 3 步才是 IP 转发的核心内容。

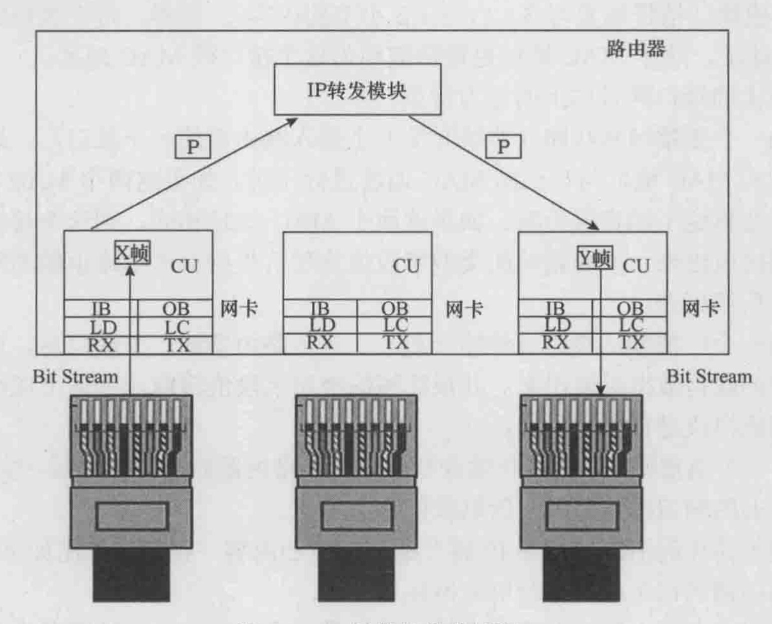


图 6-3 IP 转发的整体过程

接下来，我们以图 6-4 所示的例子来进一步地对 IP 转发的原理进行分析和描述。在这个例子中，假设 PC 1（IP 地址为 10.0.0.2/24）需要发送一个单播 IP 报文给 PC 2（IP 地址为 10.0.2.2/24）。显然，这个单播 IP 报文的源 IP 地址为 10.0.0.2/24，目的 IP 地址为 10.0.2.2/24。为简化描述，我们把这个 IP 报文取名为 P。

P 是在 PC 1 的网络层形成的。P 形成之后，PC 1 根据 P 的目的 IP 地址 10.0.2.2/24 查找自己的路由表。通过查找路由表，PC 1 知道了 P 的出接口就是自己的网口（假设 PC 1 只有一个网口），P 的下一跳 IP 地址就是路由器 A（10.0.0.0/24 网段的网关）的 Interface 1 的 IP 地址 10.0.0.1/24。

然后，PC 1 的网口会将 P 封装成一个单播帧，这个帧的载荷数据就是 P，这个帧的类型字段的值为 0x0800，这个帧的源 MAC 地址就是 PC 1 的网口的 MAC 地址，这个帧的目的 MAC 地址是 IP 地址 10.0.0.1/24 所对应的 MAC 地址。如果 PC 1 能从自己的 ARP 缓存表中查找到 IP 地址 10.0.0.1/24 所对应的 MAC 地址，则直接将这个 MAC 地址作为帧的目的 MAC 地址；否则，PC 1 的网口就会发出 ARP 请求，以便获知 IP 地址 10.0.0.1/24

所对应的 MAC 地址（请认真复习 3.4 节）。

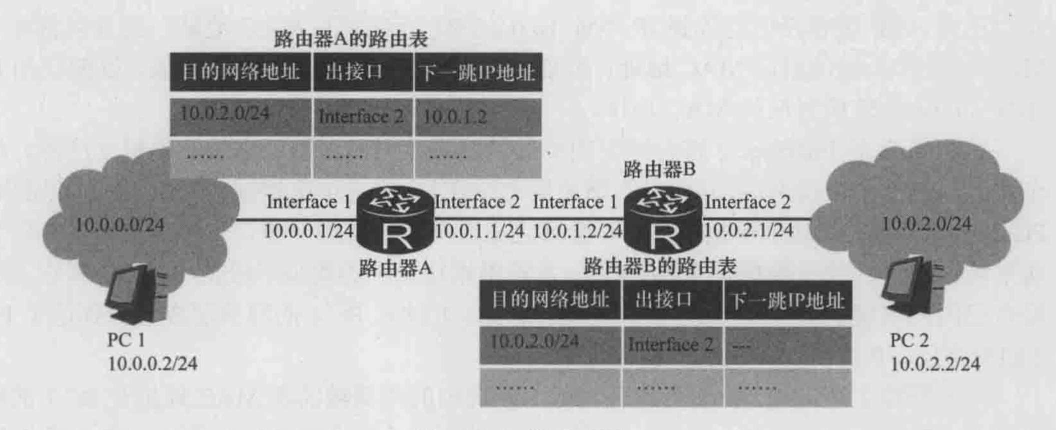


图 6-4 IP 转发示例

PC 1 的网口将封装好的单播帧发送给网云，网云中的交换机（注意，这个网云只是一个交换网络，其中没有路由器的存在）会将这个单播帧转发到路由器 A 的 Interface 1。路由器 A 的 Interface 1 接收到这个单播帧后，不会将之丢弃（请读者想一想，为什么不会丢弃），而是将这个帧的载荷数据 P 提取出来，并且根据这个帧的类型字段值 0x0800 将它上送给自己的网络层的 IP 转发模块进行处理。

路由器 A 的 IP 转发模块收到 P 后，会根据 P 的目的 IP 地址 10.0.2.2/24 去查询自己的路由表。路由表中存在这样一个条目，其含义：如果要去往 10.0.2.0/24 网段，则相应的出接口是 Interface 2，下一跳 IP 地址是 10.0.1.2/24。因为 P 的目的 IP 地址 10.0.2.2/24 是位于 10.0.2.0/24 网段的，所以 P 匹配上了这个条目。

于是，路由器 A 的 IP 转发模块会将 P 下发给 Interface 2，并告之 P 的下一跳 IP 地址是 10.0.1.2/24。Interface 2 会将 P 封装成一个单播帧，这个帧的载荷数据就是 P，这个帧的类型字段的值为 0x0800，这个帧的源 MAC 地址就是 Interface 2 的 MAC 地址，这个帧的目的 MAC 地址是 IP 地址 10.0.1.2/24 所对应的 MAC 地址。如果路由器 A 能从自己的 ARP 缓存表中查找到 IP 地址 10.0.1.2/24 所对应的 MAC 地址，则直接将这个 MAC 地址作为帧的目的 MAC 地址；否则，Interface 2 就会发出 ARP 请求，以便获知 IP 地址 10.0.1.2/24 所对应的 MAC 地址。

路由器 A 的 Interface 2 将封装好的单播帧发送出去。路由器 B 的 Interface 1 接收到这个单播帧后，不会将之丢弃，而是将这个帧的载荷数据 P 提取出来，并且根据这个帧的类型字段值 0x0800 将它上送给自己的网络层的 IP 转发模块进行处理。

路由器 B 的 IP 转发模块收到 P 后，会根据 P 的目的 IP 地址 10.0.2.2/24 去查询自己的路由表。路由表中存在这样一个条目，其含义：如果要去往 10.0.2.0/24 网段，则相应的出接口是 Interface 2，下一跳不存在，因为 Interface 2 是与 10.0.2.0/24 网段直接相连的。因为 P 的目的 IP 地址 10.0.2.2/24 是位于 10.0.2.0/24 网段的，所以 P 匹配上了这个条目。

于是，路由器 B 的 IP 转发模块会将 P 下发给 Interface 2，并告之 P 的下一跳 IP 地址不存在。Interface 2 会将 P 封装成一个单播帧，这个帧的载荷数据就是 P，这个帧的类

型字段的值为 0x0800，这个帧的源 MAC 地址就是 Interface 2 的 MAC 地址，这个帧的目的 MAC 地址就是 P 的目的 IP 地址 10.0.2.2/24 所对应的 MAC 地址。如果路由器 B 能从自己的 ARP 缓存表中查找到 IP 地址 10.0.2.2/24 所对应的 MAC 地址，则直接将这个 MAC 地址作为帧的目的 MAC 地址；否则，Interface 2 就会发出 ARP 请求，以便获知 IP 地址 10.0.2.2/24 所对应的 MAC 地址。

路由器 B 的 Interface 2 将封装好的单播帧发送给网云，网云中的交换机（注意，这个网云只是一个交换网络，其中没有路由器的存在）会将这个单播帧转发到 PC 2 的网口。PC 2 的网口接收到这个单播帧后，不会将之丢弃（请读者想一想，为什么不会丢弃？），而是将这个帧的载荷数据 P 提取出来，并且根据这个帧的类型字段值 0x0800 将它上送给自己的网络层的 IP 模块进行后续处理。至此，P 便从 PC 1 的网络层成功地到达了 PC 2 的网络层，P 的三层转发过程也告结束。

仔细回顾上述过程，我们会发现，PC 1 发送出的单播帧的源 MAC 地址是 PC 1 的网口的 MAC 地址，目的 MAC 地址是路由器 A 的 Interface 1 的 MAC 地址。路由器 A 发送出的单播帧的源 MAC 地址是路由器 A 的 Interface 2 的 MAC 地址，目的 MAC 地址是路由器 B 的 Interface 1 的 MAC 地址。路由器 B 发送出的单播帧的源 MAC 地址是路由器 B 的 Interface 2 的 MAC 地址，目的 MAC 地址是 PC 2 的网口的 MAC 地址。这说明，PC 2 所接收到的帧已经完全不是 PC 1 发送出的那个帧了（PC 1 与 PC 2 无法实现帧交换），PC 1 和 PC 2 的二层通信（数据链路层通信）是被路由器阻断了的。然而，PC 2 的网络层所接收到的 IP 报文依然是 PC 1 的网络层发出的那个 IP 报文（PC 1 与 PC 2 实现了 IP 报文交换），PC 1 与 PC 2 实现了三层通信（网络层通信）。

路由器的出现，使得网络通信领域中多了一个常用术语，这就是 internet（注意，这个词的首字母 i 是小写的）。为了解释 internet 的含义，我们不妨将图 2-4 所示的网络重新在图 6-5 中进行展示。

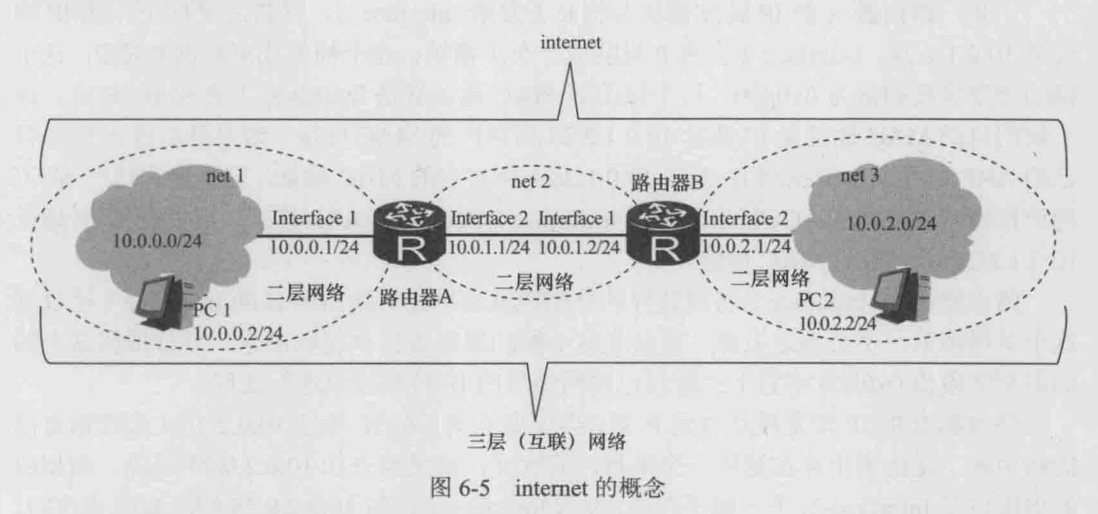


图 6-5 internet 的概念

图 6-5 所示的整个网络就是一个典型的 internet。在谈及 internet 时，我们把一个 internet 中所包含的每一个二层网络称为一个 net。所谓二层网络，就是指其中的网络接口之间总是可以进行二层（数据链路层）通信的，也就是说，其中的网络接口之间是可

以直接进行帧交换的。如果二层网络是一个以太网（即其中的网络接口都是以太口）或令牌环网（即其中的网络接口都是令牌环接口）等，则该二层网络其实就是一个二层广播域。所谓 internet，就是指由若干台（至少一台）路由器将若干个（至少两个）不同的二层网络互联而成的整体。图 6-5 所示的 internet 就是由两台路由器将三个不同的二层网络互联而成的整体。

从图 6-5 中我们可以看到，在 internet 中，路由器既是不同二层网络的分界点，又是它们的结合点，即路由器阻断了不同二层网络之间的二层通信，但却在三层（网络层）通信的层面上将不同的二层网络进行了连通。在图 6-5 中，路由器 A 的左臂 Interface 1 是属于 net 1 的，路由器 A 的右膀 Interface 2 是属于 net 2 的，左臂右膀通过路由器 A 的身体（指路由器 A 内的三层转发模块）实现了连接和互通。

在图 6-5 中，左边网云中的所有接口、PC 1 的网口以及路由器 A 的 Interface 1 是属于 net 1 的。路由器 A 的 Interface 2 和路由器 B 的 Interface 1 是属于 net 2 的。路由器 B 的 Interface 2、PC 2 的网口以及右边网云中的所有接口是属于 net 3。

有了 internet 的概念后，我们还可以对交换机和路由器进行一些比较性的描述：交换机的作用是在同一个二层网络中进行帧（Frame）的转发，而路由器的作用是在不同的二层网络之间进行包（Packet）的转发。因为帧是二层（数据链路层）数据单元，所以交换机实现的是二层转发；因为包是三层（网络层）数据单元，所以路由器实现的是三层转发。

显然，世界上存在着太多太多的 internet，因为随便买来几台路由器、交换机和电脑，就可以搭建一个独立的、属于自己的 internet。在这些数不清的 internet 中，有的规模较小，有的规模较大，其中那个规模最大的 internet 有着一个特殊的名字：Internet（注意，这个词的首字母 I 必须是大写的）。什么是 Internet？Internet 就是世界上规模最大的那个 internet，Internet 就是我们每天上网聊天所使用的那个 internet。

图 6-6 展示了规模最小的 internet 和规模最大的 internet。



图 6-6 最小的 internet 和最大的 internet

6.6 IP 报文格式

IP 报文的格式是在 IETF RFC 791 中定义的，如图 6-7 所示。

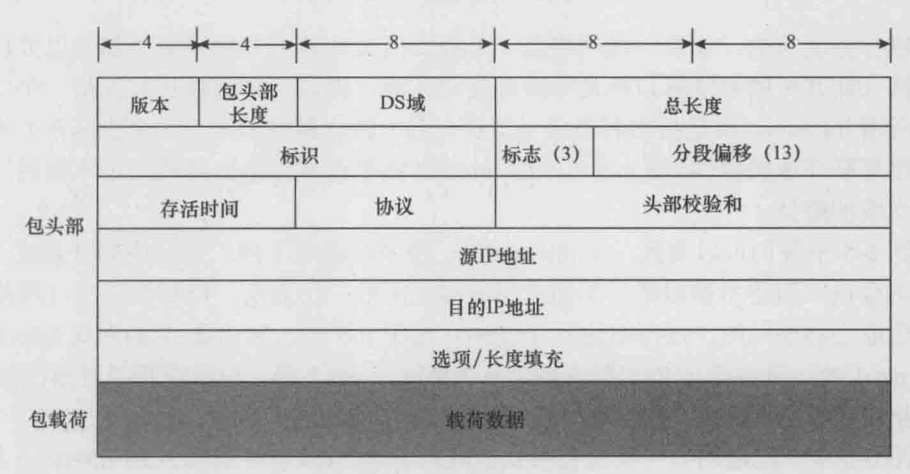


图 6-7 IP 报文格式

（1）版本

该字段长度为 4bit，表示 IP 报文的版本信息。如果该字段的值为 0x4，则表示该 IP 报文是一个 IPv4 报文。如果该字段的值为 0x6，则表示该 IP 报文是一个 IPv6 报文。注意，IPv6 报文的格式与 IPv4 报文的格式是完全不兼容的，图 6-7 所示的报文格式只是 IPv4 报文的格式，不是 IPv6 报文的格式。

（2）包头部长度

该字段长度为 4bit，用来表示 IP 包的头部的长度。由于 IP 包的头部中可能会包含一些长度不定的选项，所以 IP 包的头部的长度是不固定的（但必须是 4 字节的整数倍）。

"包头部长度"字段的值×4 = 包头部的字节数

（3）DS 域

该字段长度为 8bit，在 RFC 791 中的名称为 ToS（Type of Service）域，后来在 RFC 2474 中被重新命名为 DSCP（Differentiated Services Code Point）。该字段的作用是表示报文在 QoS（Quality of Service）中的服务等级，用以区分报文的转发优先级。

（4）总长度

该字段长度为 16bit，用来表示整个 IP 报文（IP 包的头部和 IP 包的载荷数据）的长度。一个 IP 报文的最大长度为 65 536（2^{16}）个字节。

（5）标识

该字段长度为 16bit，用于 IP 报文的分片和重组。关于 IP 报文的分片和重组，我们这里不做细究。

（6）标志

该字段长度为 3bit，用于 IP 报文的分片和重组。关于 IP 报文的分片和重组，我们这里不做细究。

（7）分段偏移

该字段长度为 13bit，用于 IP 报文的分片和重组。关于 IP 报文的分片和重组，我们这里不做细究。

（8）存活时间

该字段长度为 8bit，也称为 TTL（Time To Live）字段。当一个 IP 报文在一个 internet

中运动时，每经过一台路由器，该字段的值就被路由器减 1。如果该字段的值被减至 0，则这个报文就会被设备直接丢弃。

如果没有 TTL 机制，那么当一个 internet 中存在路由环路时，IP 报文就可能永不停止地在环路中循环运动，从而消耗大量的网络资源。有了 TTL 机制后，即使存在路由环路，IP 报文的运动时间也只能是有限的。

（9）协议

该字段长度为 8bit，用来表示 IP 报文的载荷数据的类型。例如，如果该字段的值是 0x01，则表示 IP 报文的载荷数据是一个 ICMP 报文；如果该字段的值是 0x02，则表示 IP 报文的载荷数据是一个 IGMP 报文；如果该字段的值是 0x06，则表示 IP 报文的载荷数据是一个 TCP 段；如果该字段的值是 0x11，则表示 IP 报文的载荷数据是一个 UDP 报文；如果该字段的值是 0x59，则表示 IP 报文的载荷数据是一个 OSPF 报文，如此等等。

（10）头部校验和

该字段长度为 16bit，用来对 IP 报文的头部进行差错校验。它的功能类似于以太网帧结构中的 FCS（Frame Checksum）字段（也叫 CRC 字段），但我们这里不做细究。

（11）源 IP 地址

该字段长度为 32bit，表示产生并发送该 IP 报文的设备接口的 IP 地址。

（12）目的 IP 地址

该字段长度为 32bit，表示该 IP 报文的目的接口的 IP 地址。

（13）选项/长度填充

该字段的长度是可变的。通过添加不同的选项，可以实现一些扩展功能。添加完选项之后，如果报文的头部不是 4 字节的整数倍，则必须再填充一些 0，以保证整个报文的头部长度刚好为 4 字节的整数倍。

6.7 练习题

1. （多选）以下哪些不是正确的主机接口 IP 地址？（ ）

   A．12.3.4.5.6　B．12.3.4.567　C．12.3.4.5　D．224.5.6.7

2. （多选）以下哪些地址不可用于 Internet？（ ）

   A．0.0.0.0　B．255.255.255.255　C．10.1.1.1　D．168.254.1.1

   E．172.18.1.1　F．192.1.1.1　G．192.168.1.1

3. （单选）IP 地址是 192.168.7.53，子网掩码是 255.255.255.192，则该 IP 地址所对应的网络的网络地址是？（ ）

   A．192.168.7.0　B．192.168.7.128　C．192.168.7.192　D．192.168.7.224

4. （单选）IP 地址是 192.168.7.53，子网掩码是 255.255.255.192，则该 IP 地址所对应的网络的广播地址是？（ ）

   A．192.168.7.255　B．192.168.7.127

   C．192.168.7.63　D．192.168.7.31

5. （单选）192.168.7.53/18 所在网络中的可用主机接口 IP 地址有多少个？（ ）

A．256　　B．254　　C．16 384　　D．16 382

E．262 144　　F．262 142

6．（多选）以下描述中错误的是？（　）

A．路由器的接口收到一个广播帧后，会把这个广播帧直接丢弃，不进行任何三层处理

B．路由器的接口收到一个广播帧后，会把这个广播帧进行泛洪

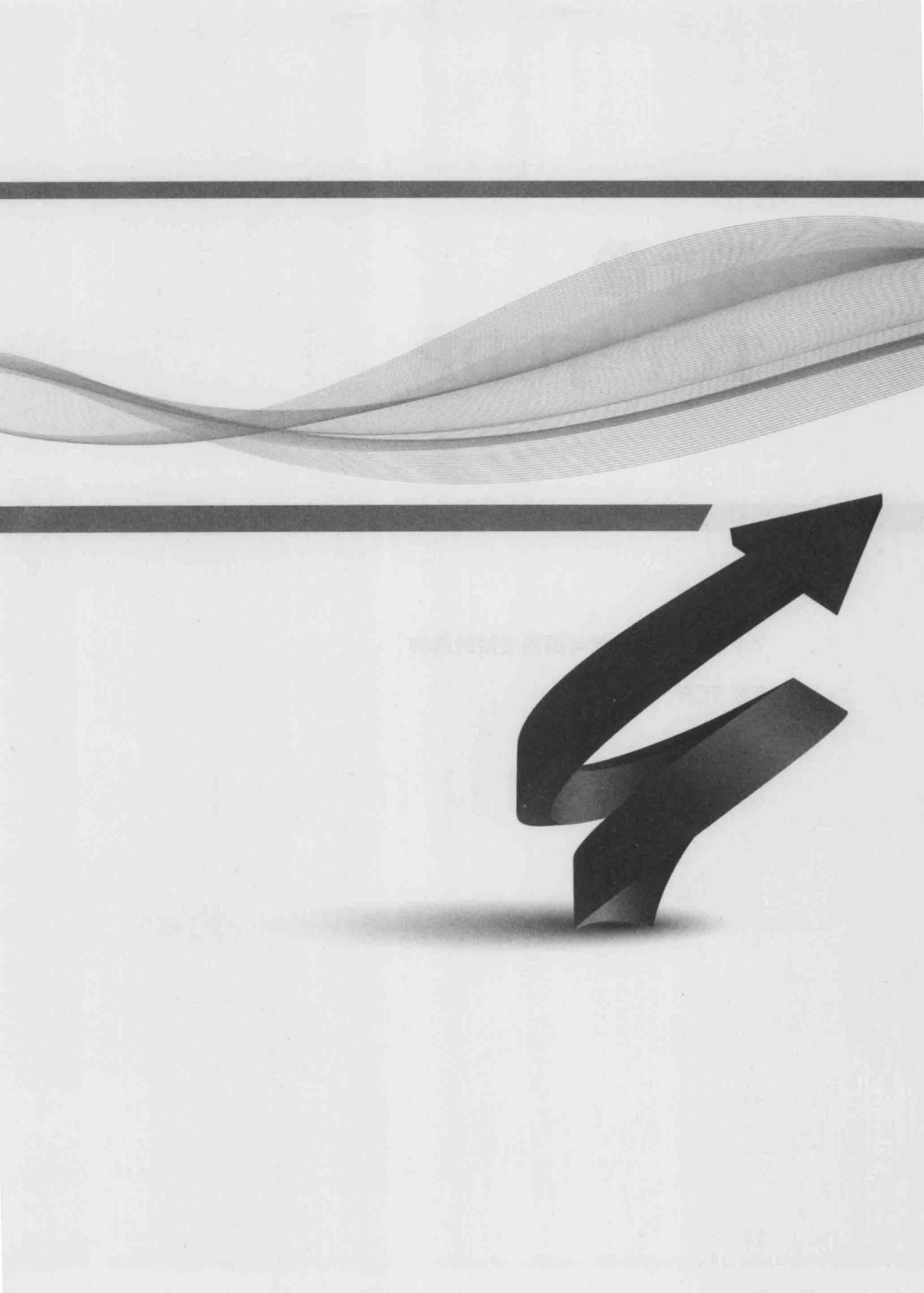
C．路由器的接口收到一个单播帧后，可能会把这个帧直接丢弃

7．（多选）以下关于 IP 报文的说法中正确的有？（　）

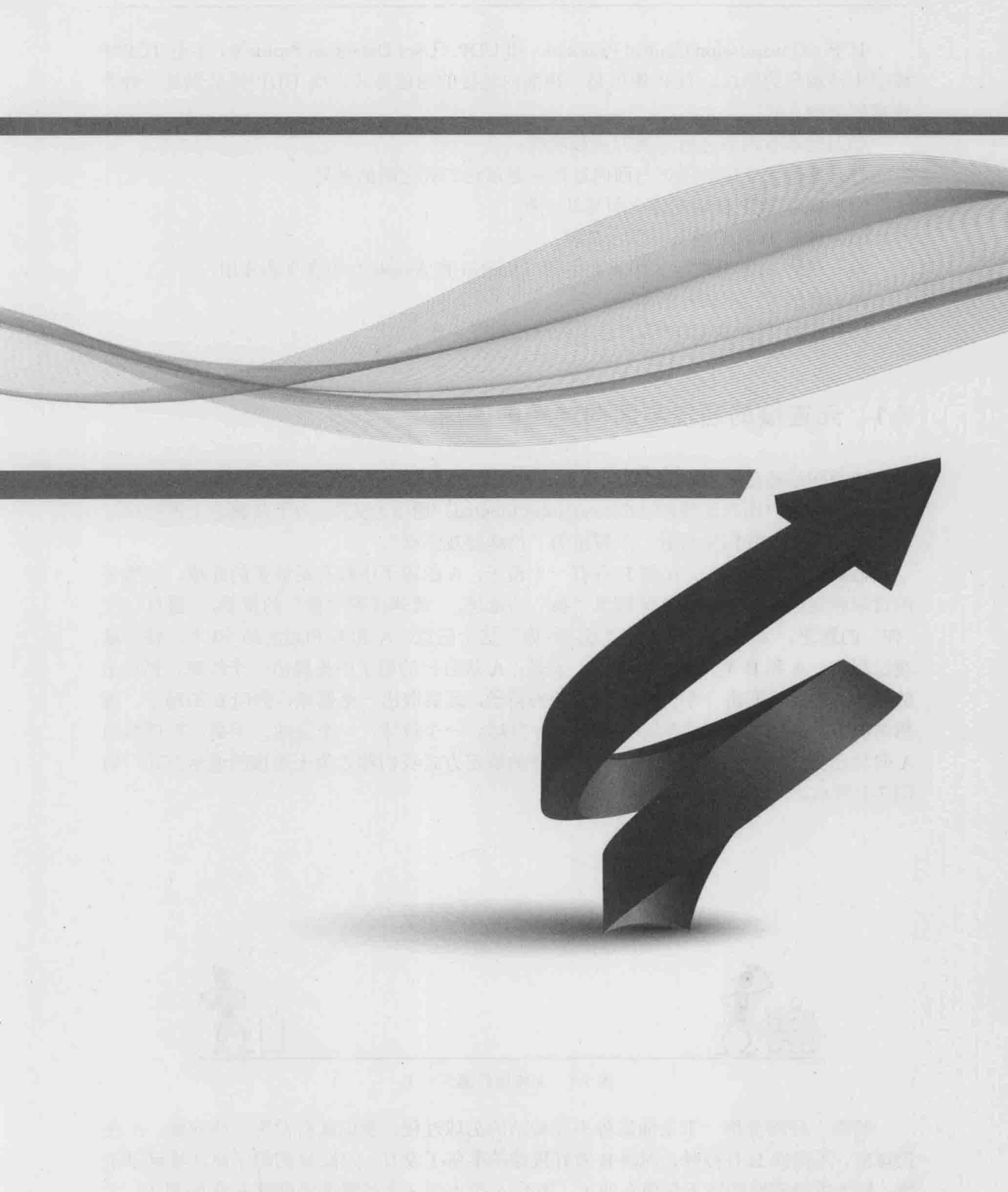
A．IP 报文没有尾部

B．IP 报文的长度不能超过 65 536 字节

C．IP 报文头部至少有 20 字节



第7章 TCP与UDP



TCP（Transmission Control Protocol）和 UDP（User Datagram Protocol）都是 TCP/IP 模型中传输层的协议。TCP 通信是一种面向连接的通信方式，而 UDP 通信则是一种非连接的通信方式。

学习完本章内容之后，我们应该能够：

（1）理解无连接的通信与面向连接的通信这二者之间的差异；

（2）理解 TCP 会话的建立和终结过程；

（3）理解 TCP 的确认与重传机制；

（4）理解 TCP 分段格式中重要字段（SeqNo 和 AckNo）的含义和作用；

（5）理解应用端口号的作用；

（6）理解 TCP 和 UDP 各自适合的应用场景。

7.1 无连接的通信与面向连接的通信

谈及网络通信时，我们经常会听说两种不同的通信方式："无连接的（Connectionless）通信方式"和"面向连接的（Connection-Oriented）通信方式"。为了从概念上理解这二者之间的差异，我们先来看一个假想的"扔球游戏活动"。

假设有 A、B 两人，A 和 B 各有一个箱子，A 的箱子中装有足够多的红球、足够多的黄球和足够多的篮球。红球代表"我"的意思， 黄球代表"爱"的意思， 篮球代表"你"的意思，A 希望向 B 传递"我-爱-你"这个信息。A 和 B 相距大约 50 米，整个游戏过程中，A 和 B 不允许相互喊话。于是，A 从自己的箱子中先取出一个红球，扔向 B 的箱子，然后再取出一个黄球，扔向 B 的箱子，最后取出一个篮球，扔向 B 的箱子。理想情况下，B 的箱子中会顺序地落入一个红球、一个黄球、一个蓝球，于是，B 便明白 A 向自己传递的信息是"我-爱-你"。这样的通信方式我们称之为无连接的通信方式，如图 7-1 所示。

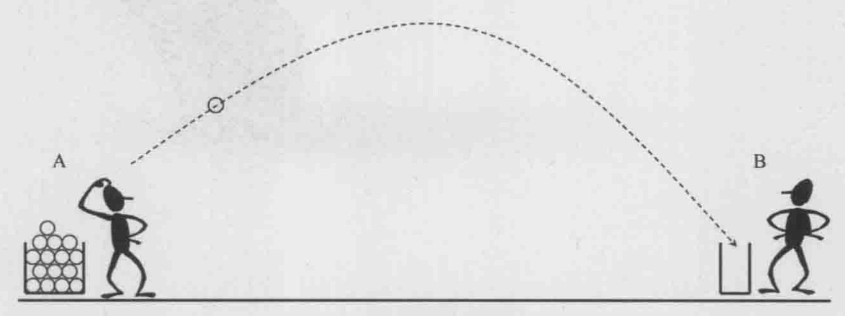


图 7-1 无连接的通信方式

然而，仔细分析一下上面这种不许喊话的扔球过程，我们就会发现一些问题。A 在扔球前，不能向 B 打招呼，询问 B 做好接球的准备了没有。如果 B 的箱子里还堆满了杂物，根本就没有空间装下任何东西了，那么 A 扔出的 3 个球就都不能落入 B 的箱子，于是 B 完全接收不到"我-爱-你"这个信息。

即使B的箱子一开始有足够的空间装下足够数量的球，但是A扔出的第二个球（黄球）被突然刮起的一阵强风吹偏了，结果只有红球和蓝球顺序地落入了B的箱子，于是B接收到的信息就是“我-你”。显然，这个信息并非是A希望传递的信息。

也可能出现这样的情况，A虽然是按红球、黄球、蓝球这样的顺序将球扔出去的，但是由于扔球的角度差异和力度差异以及风力、风向的改变等因素，最终落入B的箱子的顺序变成了蓝球、黄球、红球。这样一来，B接收到的信息就是“你-爱-我”，而非“我-爱-你”。

事实上，还有很多其他的意外情况，导致B不能最终正确地接收到“我-爱-你”这个信息。A顺序地扔完了红球、黄球、蓝球后，并不能确定这3个球是否全部落入了B的箱子，也不确定落入的顺序对不对，因为A是不能从B那里得到任何反馈信息的。B虽然可以知道球落入自己箱子的顺序，但是却无法知道落入的顺序是否就是A扔出的顺序。另外，B也无法知道A总共要扔几个球，而且也无法知道自己是否收漏了什么球没有。总而言之，B最终能否顺序地接收到一个红球、一个黄球、一个蓝球，从而正确地接收到A希望传递的信息，很大程度上就只能凭“运气”了。从这个意义上讲，我们就说无连接的通信方式是一种不可靠的通信方式。

为了提高信息传递的可靠性，我们现在允许A和B可以相互喊话（但喊话的内容不允许涉及球的颜色）以及采取一些其他的措施来控制整个扔球的过程，如图7-2所示。例如，A在扔出每一个球之前，都会在这个球上写上序号：若是红球就写5，若是黄球就写6，若是蓝球就写7。A在最开始扔球之前，必须先向B喊话：“你做好接球的一切准备工作了吗？我所扔出的球的开始序号是5”。如果A等了一会儿后，还听不到B的回话，就会再喊一遍。如果A喊了三遍都听不到B的回话（比如，B根本就没有到达游戏活动的现场），A就认为游戏没法开始进行，于是什么球都不扔，游戏就算结束了。当然，通常情况下，B会回应道：“我已经准备好了，我已知道你扔的第一个球将会是5号球，你开始扔吧”。A听到后，又会回应道：“好的，那我开始扔了”。然后，A就开始顺序地扔出一个红球、一个黄球、一个蓝球。在这个过程中，A和B需要一直保持相互喊话的状态，以控制整个扔球和接球的过程。比如，B在收到了第一个球后，发现自己的箱子没有多余的空间装下更多的球，就会立即向A喊道：“后面的球请等会儿再扔，我需要一点时间把箱子腾出空来”。又比如，B可能会向A喊道：“5号球和6号球我都收到了，你可以扔后面的球了”。也有这样的可能，B已经接收到了5号球和7号球，但是一直没收到6号球，于是B会向A喊道：“我还没有收到6号球，请你再扔一次6号球”。A听见后，就会再扔一个写有6号的黄球，然后向B喊到：“你收到6号球后请告诉我一下”，如此等等。A和B一直相互喊话，以便控制扔球和接球的过程。最后，A从B的喊话中得知，B已经收到了5、6、7号球，于是A向B喊到：“好的，我的球扔完了，我准备结束游戏了”。B听到后，回应到：“好的，那就结束吧”。于是，游戏结束。然后，B按照序号从小到大（增量为1）的顺序来理解所接收到的3个球的意思（注意，这3个球落入箱子的顺序并不一定是红、黄、蓝），结果一定会是“我-爱-你”。

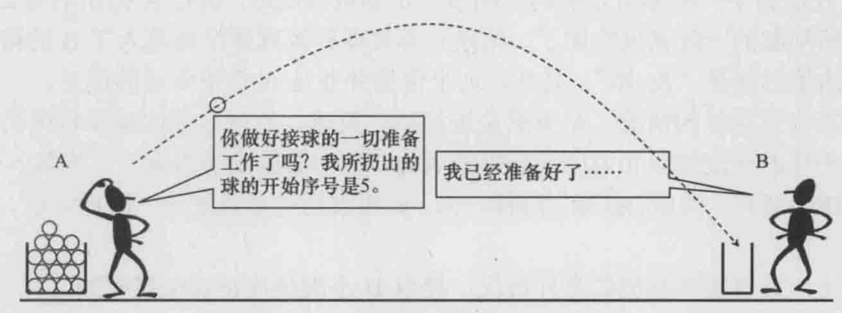


图 7-2 面向连接的通信方式

前面这种可以相互喊话的扔球方式便是一种面向连接的通信方式。显然，面向连接的通信方式是一种可靠的通信方式。A、B 双方通过相互喊话的机制，控制了扔球活动何时开始、何时结束，同时也对扔球活动的中间过程进行了严格的控制。比如，B 如果觉得 A 每扔一个球之间的时间隔得太短了，则可以通过喊话让 A 的动作慢一点；B 如果觉得 A 每扔一个球之间的时间隔得太长了，则可以通过喊话让 A 的动作快一点。由于每个球都有序号，B 就能够发现自己是否收漏了什么球没有，并且通过喊话让 A 知道自己有哪些序号的球没有收到，以便让 A 重新把这些球扔过来。A 也会不停地向 B 喊话，以确认哪些球 B 已经收到了，哪些球 B 还没有收到。通过这种相互喊话的控制机制，B 自然可以非常可靠地接收到 A 希望传递的信息。

如果以网络通信的眼光来看待喊话式的扔球活动，则每一个球就可以被称为是一个“数据报文”，而喊话内容中的每一个字就可以被称为是一个“控制报文”。A、B 双方通过交互“控制报文”，严格地控制了“数据报文”的整个传递过程。从 A 在最开始扔球之前向 B 喊话：“你做好接球的一切准备工作了吗？……”，一直到 B 最后说：“好的，那就结束吧”，整个过程被称为一次“通信会话”，或简称为“会话（Session）”。显然，A、B 双方是通过“控制报文”的交互而实现并控制了“会话”的开始、结束，以及“会话”的中间过程。

理解了无连接的通信方式及面向连接的通信方式这两个概念后，我们来分析一下二层通信（这里指以太网通信）。在一个二层网络内部，不同的接口之间是通过帧（Frame）交换的方式来相互传递信息的。我们知道，发送方的网卡在向接收方的网卡发送帧之前，不会以任何方式通知接收方的网卡做好接收准备。发送方的网卡也无法确定接收方的网卡是否按顺序接收到了自己所发出的所有的帧。由于帧的结构中是没有帧的序号信息的，所以接收方的网卡无法知道自己收漏了什么帧没有。接收方的网卡即使知道自己收漏了帧，也无法通知发送方的网卡进行重新发送。总之，发送方的二层功能模块与接收方的二层功能模块之间缺乏任何控制机制来控制它们之间的帧交换过程。因此，二层通信（这里指以太网通信）是一种无连接的通信方式。

我们再来分析一下 internet 上进行的三层通信（网络层通信，IP 通信）。三层通信时，发送方的 IP 模块与接收方的 IP 模块是通过包（IP Packet）交换的方式来相互传递信息的。我们知道，发送方的 IP 模块在向接收方的 IP 模块发送包之前，不会以任何方式通知接收方的 IP 模块做好接收准备。发送方的 IP 模块也无法确定接收方的 IP 模块是否按

顺序接收到了自己所发出的所有的包。由于IP包的结构中是没有包的序号信息的，所以接收方的IP模块无法知道自己收漏了什么包没有。接收方的IP模块即使知道自己收漏了包，也无法通知发送方的IP模块进行重新发送。总之，发送方的位于三层的IP模块与接收方的位于三层的IP模块之间缺乏任何控制机制来控制它们之间的包交换过程。因此，三层通信（网络层通信，IP通信）是一种无连接的通信方式。

7.2 TCP

从图1-7我们可以知道，TCP（Transmission Control Protocol）协议是TCP/IP协议簇中传输层的一个协议。

在网络通信中，数据链路层（二层）上会进行帧（Frame）的交换，网络层（三层，IP层）上会进行IP包（IP Packet）的交换，在传输层（四层）上还可能会进行TCP段（TCP Segment）的交换。

在发送方，TCP模块接收到应用层下送的数据之后，会将这些数据封装成TCP段，这些段被称为TCP数据段。发送方的TCP模块在发送这些TCP数据段之前（也就是将这些TCP数据段下送给三层的IP模块之前），必须首先向接收方的TCP模块发送一些TCP控制段，然后通过与接收方的TCP模块进行TCP控制段的交互而建立起TCP会话（TCP Session）。TCP会话建立之后，发送方的TCP模块和接收方的TCP模块之间才开始进行TCP数据段的传递。在TCP数据段的传递期间，发送方的TCP模块和接收方的TCP模块之间会一直保持TCP控制段的交互。最后，发送方的TCP模块和接收方的TCP模块之间会通过TCP控制段的交互来终止TCP会话，从而结束TCP数据段的传递。显然，TCP通信是一种面向连接的、可靠的通信方式。

我们前面说到，二层的帧通信和三层的包通信都是无连接的、不可靠的通信方式。幸运的是，四层的TCP通信却是一种可靠的通信方式。如果帧（Frame）在传递过程中被搞丢了，通信双方的二层功能模块都是发现不了的；如果包（IP Packet）在传递过程中被搞丢了，通信双方的三层IP模块也是发现不了的。然而，如果一个TCP段被搞丢了，则TCP模块是一定能够发现的。一个TCP段的丢失，就意味着一个IP包的丢失（因为TCP段是作为载荷数据封装在IP包中的）；一个IP包的丢失，就意味着一个帧的丢失（因为IP包是作为载荷数据封装在帧中的）。这样一来，二层通信和三层通信的不可靠性在TCP这里便得到了补偿。

7.2.1 TCP会话的建立

如图7-3所示，假设计算机A是希望向计算机B发起通信的一方，计算机A的TCP模块与计算机B的TCP模块之间将通过“三次握手（Three-way Handshaking）”来建立TCP会话的。

所谓三次握手，是指在TCP会话的建立过程中总共交换了3个TCP控制段，它们分别是一个SYN段、一个SYN+ACK段、一个ACK段。SYN是Synchronization的缩写，ACK是Acknowledgement的缩写。

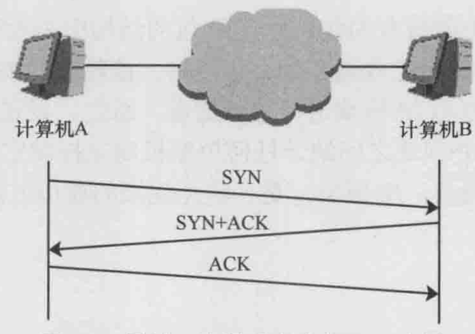


图 7-3 通过 3 次握手来建立 TCP 会话

SYN 段是由计算机 A 的 TCP 模块产生并发送的，用于向计算机 B 的 TCP 模块发起建立会话的请求，同时也把自己的状态告诉对方。SYN+ACK 段是由计算机 B 的 TCP 模块发送的回应，同时也将自己的状态告诉给计算机 A 的 TCP 模块。ACK 段是计算机 A 的 TCP 模块对计算机 B 的 TCP 模块的回应作出的回应，用于确认会话的建立。

需要特别注意的是，经过 3 次握手之后，A、B 之间其实是建立起了两个 TCP 会话，一个是从 A 指向 B 的 TCP 会话，另一个是从 B 指向 A 的 TCP 会话。因为 A 是发起通信的一方，说明 A 有信息要传递给 B，于是 A 首先发送了一个 SYN 段，请求建立一个从 A 指向 B 的 TCP 会话，这个会话的目的是要控制信息能够正确而可靠地从 A 传递给 B。B 在收到 SYN 段后，会发送一个 SYN+ACK 段作为回应。SYN+ACK 段的含义是：B 一方面同意了 A 的请求，另一方面也请求建立一个从 B 指向 A 的 TCP 会话，这个会话的目的是要控制信息能够正确而可靠地从 B 传递给 A。A 收到 SYN+ACK 段后，回应一个 ACK 段，表示同意 B 的请求。

7.2.2 TCP 会话的终止

计算机 A 的 TCP 模块和计算机 B 的 TCP 模块经过 3 次握手建立起了 TCP 会话之后，就可以开始进行 TCP 数据段的交换了。在 TCP 数据段的交换期间，A 和 B 的 TCP 模块之间会同时保持 TCP 控制段的交互。当 TCP 数据段的交换结束时，双方需要相互发送 FIN（FIN 是 Finish 的缩写）段和 ACK 段这样的 TCP 控制段来明确地终止 TCP 会话，这种方式称为“4 次握手（Four-way Handshaking）”，如图 7-4 所示。

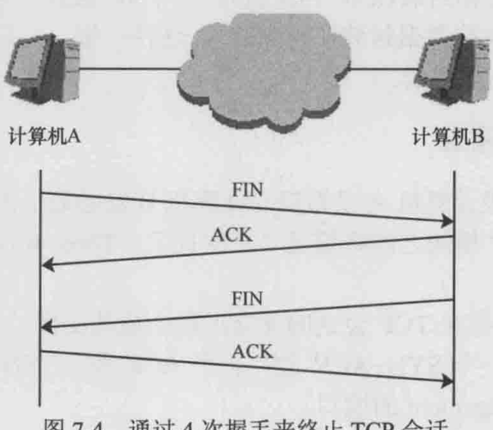


图 7-4 通过 4 次握手来终止 TCP 会话

从图 7-4 我们可以看到，TCP 会话的终止分为两个部分。首先，A 发出一个 FIN 控制段，表示请求终止从 A 指向 B 的 TCP 会话。B 回应一个 ACK 段，表示同意 A 的请求，然后就开始进行终止这个会话的操作。A 在收到 B 回应的 ACK 段后，才开始进行终止这个会话的操作。另一方面，B 也向 A 发出一个 FIN 段，表示请求终止从 B 指向 A 的 TCP 会话。A 回应一个 ACK 段，表示同意 B 的请求，然后就开始进行终止这个会话的操作。B 在收到 A 回应的 ACK 段后，才开始进行终止这个会话的操作。

7.2.3 TCP 段的格式

一个 TCP 段也经常被称为一个 TCP 分段，图 7-5 显示了 TCP 分段的格式。

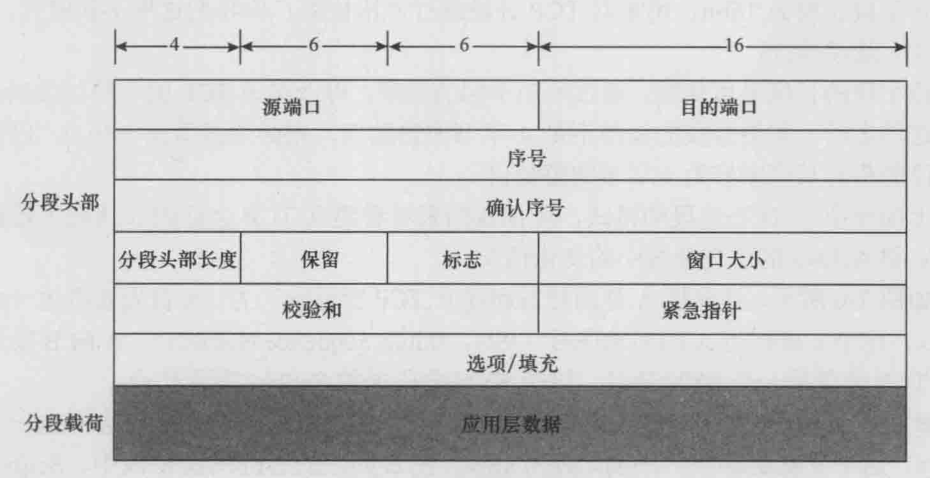


图 7-5 TCP 分段的格式

关于 TCP 分段的格式，我们只需要了解其中部分字段的含义和作用。对于其他一些字段的理解和认识，已经超出了本书的知识范围。

（1）源端口

该字段也称为源端口号，长度为 16bit，用来表示该 TCP 分段的载荷数据是应用层的哪个应用模块产生并发送的。

（2）目的端口

该字段也称为目的端口号，长度为 16bit，用来表示该 TCP 分段的载荷数据应该由应用层的哪个应用模块来接收并处理。

（3）序号

该字段的英文名称是 Sequance Number，简写为 SeqNo。该字段的长度为 32bit，它是该 TCP 分段自身的序号。接收这个分段的一方可以根据这个序号来判断是否存在分段重收或漏收等情况。

（4）确认序号

该字段的英文名称是 Acknowledgement Number，简写为 AckNo。其含义会在后续的例子中得到解释。

（5）分段头部长度

该字段的长度为 4bit，它标识了分段头部的长度。由于分段头部中可能包含一些选项，所以分段头部的长度是不固定的，但分段头部的字节数必须是 4 的整数倍。

“分段头部长度”字段的值×4＝ 分段头部的字节数

（6）标志

该字段包含了 6 个比特，其中的每个比特都有自己的名称和含义。这 6 个比特分别是：URG、ACK、PSH、RST、SYN、FIN。对于 ACK 分段和 ACK+SYN 分段，ACK 比特应该置 1。对于 SYN 分段和 ACK+SYN 分段，SYN 比特应该置 1。对于 FIN 分段，FIN 比特应该置 1。

（7）校验和

该字段长度为 16bit，用来对 TCP 分段进行差错校验，但我们这里不做细究。

（8）选项/填充

该字段的长度是可变的。通过添加不同的选项，可以实现 TCP 的一些扩展功能。添加完选项之后，如果分段的头部不是 4 字节的整数倍，则必须再填充一些 0，以保证整个分段的头部长度刚好为 4 字节的整数倍。

上面介绍了 TCP 分段的格式，现在我们来看看建立 TCP 会话的 3 次握手过程中，SeqNo 和 AckNo 的值在分段中的变化情况。

如图 7-6 所示，计算机 A 是最初发起建立 TCP 会话的一方。A 首先要产生一个随机整数 x，这个 x 被称为 A 的初始序号（ISN，Initial Sequence Number）。A 向 B 发送的第一个 TCP 分段是一个 SYN 分段，这个 SYN 分段中的 SeqNo 就等于 x。

B 在收到 A 发送的 SYN 分段后，会回应一个 SYN+ACK 段。B 也会产生一个随机整数 y，这个 y 就是 B 自己的初始序号 ISN。在 B 回应的 SYN+ACK 段中，SeqNo 的值就是 y。同时，SYN+ACK 段中 AckNo 的值为 x 的值加上 1。

最后，A 回应一个 ACK 分段给 B。在这个 ACK 分段中，SeqNo 的值为 x 的值加上 1，AckNo 的值为 y 的值加上 1。至此，TCP 会话的建立过程便告结束。

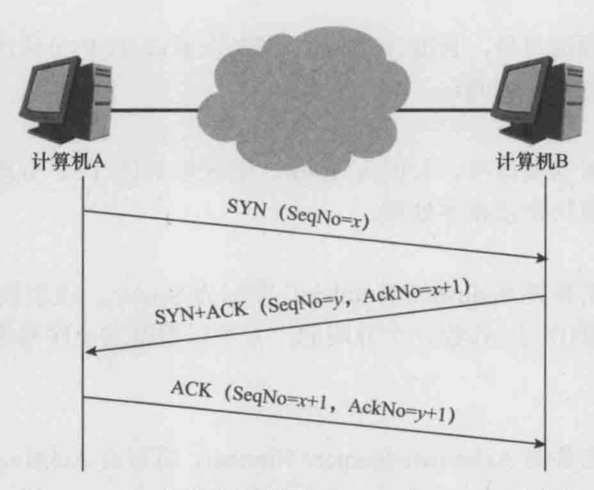


图 7-6　TCP 会话建立过程中 SeqNo 和 AckNo 的变化

7.2.4 TCP的确认与重传机制

我们将继续以图 7-6 所示的例子来描述 TCP 的一个非常重要的机制：确认与重传机制。

假设 A 与 B 建立了 TCP 会话之后，A 有很多个 TCP 数据段需要传递给 B。假设 A 需要传递的第一个数据段的长度是 400 字节，第二个数据段的长度是 500 字节，第三个数据段的长度是 800 字节。另外，假设 A 的初始序号 x 为 1 367。

A 完成了第三次握手（向 B 回应了一个 ACK 分段）之后，便可以开始发送第一个数据段。A 在发送第一个数据段时，数据段的 SeqNo 应该为 1 369，因为 1 367 和 1 368 已经在三次握手过程中被 A 使用过了。注意，1 369 其实是第一个数据段的第一个字节的序号。第一个数据段的最后一个字节的序号是 1 768（1 369＋400－1＝1 768）。A 在发送完第一个数据段后，便开始等待来自 B 的确认信息。

B 在成功接收到了第一个数据段后，会向 A 回应一个 ACK 段。这个 ACK 段的 AckNo 的值为 1 769（1 369＋400＝1 769），它的含义是：我希望下次开始接收序号为 1 769 的字节，因为我已经接收到了序号为 1 768 之前的所有字节。

A 收到 B 回应的 ACK 段后，就开始发送第二个数据段。发送第二个数据段时，数据段的 SeqNo 为 1 769。注意，1 769 其实是第二个数据段的第一个字节的序号。第二个数据段的最后一个字节的序号是 2 268（1 769＋500－1＝2 268）。A 在发送完第二个数据段后，又开始等待来自 B 的确认信息。

B 在成功接收到了第二个数据段后，又会向 A 回应一个 ACK 段。这个 ACK 段的 AckNo 的值为 2 269（1 769＋500＝2 269），它的含义是：我希望下次开始接收序号为 2 269 的字节，因为我已经接收到了序号为 2 268 之前的所有字节。

A 收到 B 回应的 ACK 段后，就开始发送第三个数据段。发送第三个数据段时，数据段的 SeqNo 为 2 269。注意，2 269 其实是第三个数据段的第一个字节的序号。第三个数据段的最后一个字节的序号是 3 068（2 269＋800－1＝3 068）。A 在发送完第三个数据段后，又开始等待来自 B 的确认信息。

B 在接收到了第三个数据段后，又会向 A 回应一个 ACK 段。这个 ACK 段的 AckNo 的值为 3 069（2 269＋800＝3 069），它的含义是：我希望下次开始接收序号为 3 069 的字节，因为我已经接收到了序号为 3 068 之前的所有字节。

A 收到 B 回应的 ACK 段后，就开始发送第四个数据段。发送第四个数据段时，数据段的 SeqNo 为 3 069……图 7-7 示意了上面描述的过程。

假如，A 发送的第二个数据段在传递过程中丢失了（比如，A 发出的某个帧在传递过程中丢失了，这个帧封装了一个 IP 包，而这个 IP 包封装了这个数据段），那么 B 就不可能向 A 回应相应的 ACK 段。因为 A 没有接收到 B 对这个数据段的回应，所以 A 就不能开始发送第三个数据段。A 能做的事情就是继续等待 B 对第二个数据段的回应。当 A 的等待时间超时后，A 就会认为 B 没有成功接收到第二个数据段。于是，A 就会重新发送第二个数据段，然后重新等待 B 的回应。图 7-8 示意了这一过程。



图 7-7　TCP 的确认与重传机制示意一

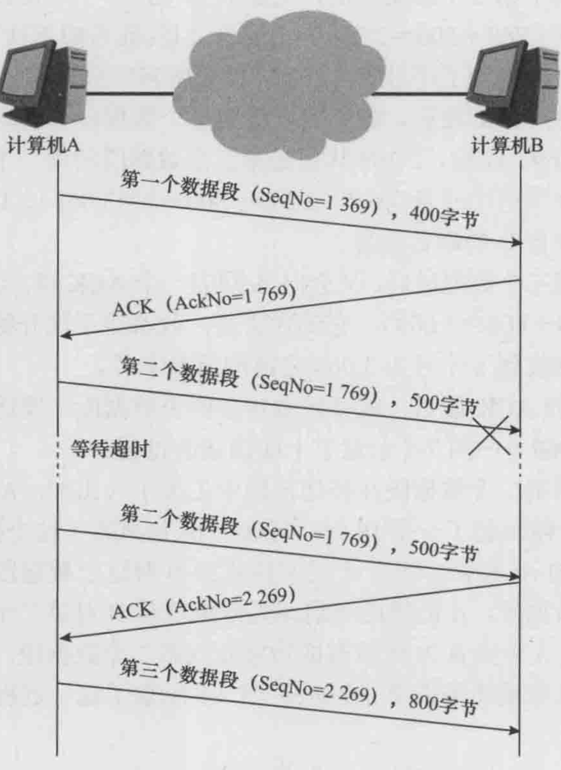


图 7-8　TCP 的确认与重传机制示意二

上面的例子已经清晰地展示了确认与重传机制的基本特征。然而，我们也可以发现，在这个例子中，A 每发送一个数据段，都必须等待 B 的一个确认（回应），所以传输效率很低。为了解决这个问题，TCP 协议还提供了一种被称为“滑动窗口（Sliding Window）”的机制来提高传输效率。关于滑动窗口机制，我们这里不做描述和讨论。事实上，除了滑动窗口机制外，TCP 还存在很多丰富而精细的特性，这些内容都有待读者朋友们以后去学习和探索。

7.2.5　应用端口

在描述 TCP 分段格式的时候，我们提到了源端口和目的端口。这里所说的端口，并不是设备的物理端口，而是一种抽象的、被称为“应用端口（Application Port）”的端口。TCP 分段中的应用端口的作用是标识 TCP 段的载荷数据对应了哪个应用层的模块。

应用端口分为两类：知名端口（Well-known Port）和非知名端口。知名端口的编号范围是 0~1 023，这个范围的端口号被标准组织专门用来分配给一些特定的应用层模块，以保证所有的网络设备都可以正确识别 TCP 分段中载荷数据所属的应用类型。表 3-1 中给出了几个知名端口的例子。非知名端口号的范围是 1 024~65 535，这个范围内的端口号没有固定的使用场合，由网络设备在通信时动态分配给需要通信的应用程序。

表 7-1　　知名 TCP 端口号示例

端口号	应用	说明
20	FTP 数据	FTP（File Transfer Protocol，文件传输协议）用于在两台网络设备之间传输文件。FTP 使用 20 号端口来传递文件数据，使用 21 号端口来传递控制数据
21	FTP 控制	
23	Telnet	用于通过远程方式来控制网络设备
25	SMTP（Simple Mail Transfer Protocol，简单邮件传输协议）	用于发送电子邮件
53	DNS（Domain Name System，域名解析系统）	用于在 IP 地址和便于记忆的域名之间进行自动转换
80	HTTP（Hypertext Transfer Protocol，超级文本传输协议）	用于浏览网站和网页
110	POP3（Post Office Protocol 3，第三版邮局协议）	用于接收电子邮件

7.3　UDP

在网络通信中，信息传递的可靠性与信息传递的效率之间总是一对难解的矛盾。有的情况下，我们需要损失信息传递的效率来增强信息传递的可靠性；有的情况下，我们需要损失信息传递的可靠性来提升信息传递的效率。

TCP 的确认与重传机制，保证了信息的可靠传递，但同时也或多或少地降低了信息传递的效率。实际上，随着网络技术的发展，传输介质的传输速度和抗干扰能力越来越高，网络设备也越来越稳定和可靠，信息传递出现错误的概率已经非常小了。另一方面，

有一些网络应用对于信息传递的可靠性的要求并不是那么高。比如，我们在线观看视频时，如果其中只是个别的画面在传输过程中发生了丢失，我们的肉眼是根本感觉不出来的。在这种情况下，如果我们的电脑自动去要求视频的发送方重新发送这些丢失的画面，反而会让我们等得不耐烦。

在传输层上，我们还有一个协议，称为 UDP（User Datagram Protocol）。UDP 通信是一种非连接的通信方式。

如图 7-9 所示，UDP 报文（UDP Datagram）的格式非常简单。UDP 报文头部中，源端口和目的端口的含义与 TCP 分段头部中的源端口和目的端口的含义是完全一样，其他字段的含义也是显而易见的，这里不再赘述。顺便提醒一下，UDP 报文的头部中不存在任何字段来表示报文的序号。

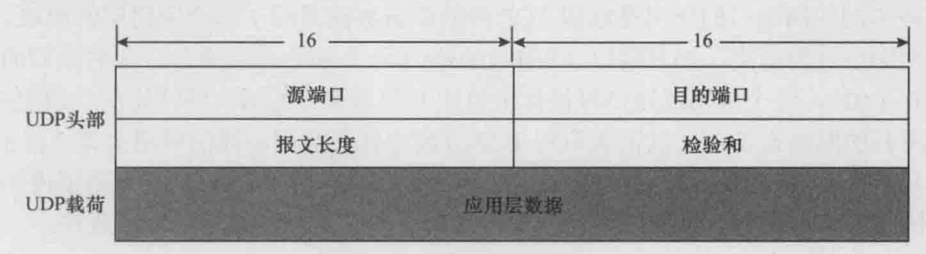


图 7-9 UDP 报文的格式

当然，UDP 报文也没有数据报文和控制报文之分，所有的 UDP 报文都是 UDP 数据报文。信息发送方的 UDP 模块与信息接收方的 UDP 模块之间也不存在 UDP 会话的概念，也没有所谓的确认与重传机制。发送方的 UDP 模块将应用层下发的数据封装成 UDP 报文后，会将 UDP 报文直接下送给三层的 IP 模块。接收方的 UDP 模块在收到三层的 IP 模块上送的 UDP 报文后，会将 UDP 报文中的载荷数据上送给目的端口号所对应的应用层模块。

UDP 协议之所以省去了 TCP 协议所做的大量工作，包括处理报文的丢失、重复、时延、乱序等各种问题，是因为 UDP 认为信息的可靠性传输可以由应用层来提供保证。如果应用程序需要高可靠性的信息传递，那么程序自身可以自带确认与重传等机制。如果应用程序不需要那么高的信息传递的可靠性，那么应用程序也可以省去这些功能机制。

最后的问题是，对于某种应用程序来说，究竟是该用 TCP 作为它的传输层协议呢，还是该用 UDP 作为它的传输层协议？答案是 IETF 已经规定了哪些应用程序必须使用 TCP；哪些应用程序必须使用 UDP；哪些应用程序既可以使用 TCP，也可以使用 UDP；哪些应用程序既不使用 TCP，也不使用 UDP，而是跳过传输层直接使用 IP 协议。对这种对应关系感兴趣的读者可以去查阅相关的标准文档。

7.4 练习题

1. （单选）UDP 的全称是？（ ）

A．User Delivery Protocol　　B．User Datagram Procedure

C．User Datagram Protocol　　D．Unreliable Datagram Protocol

2．（单选）TCP 模块在发送 TCP 分段时，其实是将 TCP 分段交给？（　）

A．应用层　B．传输层　C．网络层　D．数据链路层

3．（多选）下列整数中，可以成为一个 UDP 端口号的是？（　）

A．1　B．80　C．2 048　D．65 536

4．（多选）TCP 协议的三次握手过程中不会涉及到下面哪些 TCP 分段？（　）

A．SYN　B．SYN+ACK　C．ACK

D．FIN　E．FIN+ACK

5．（多选）TCP 会话终止的过程中会涉及到下面哪些 TCP 分段？（　）

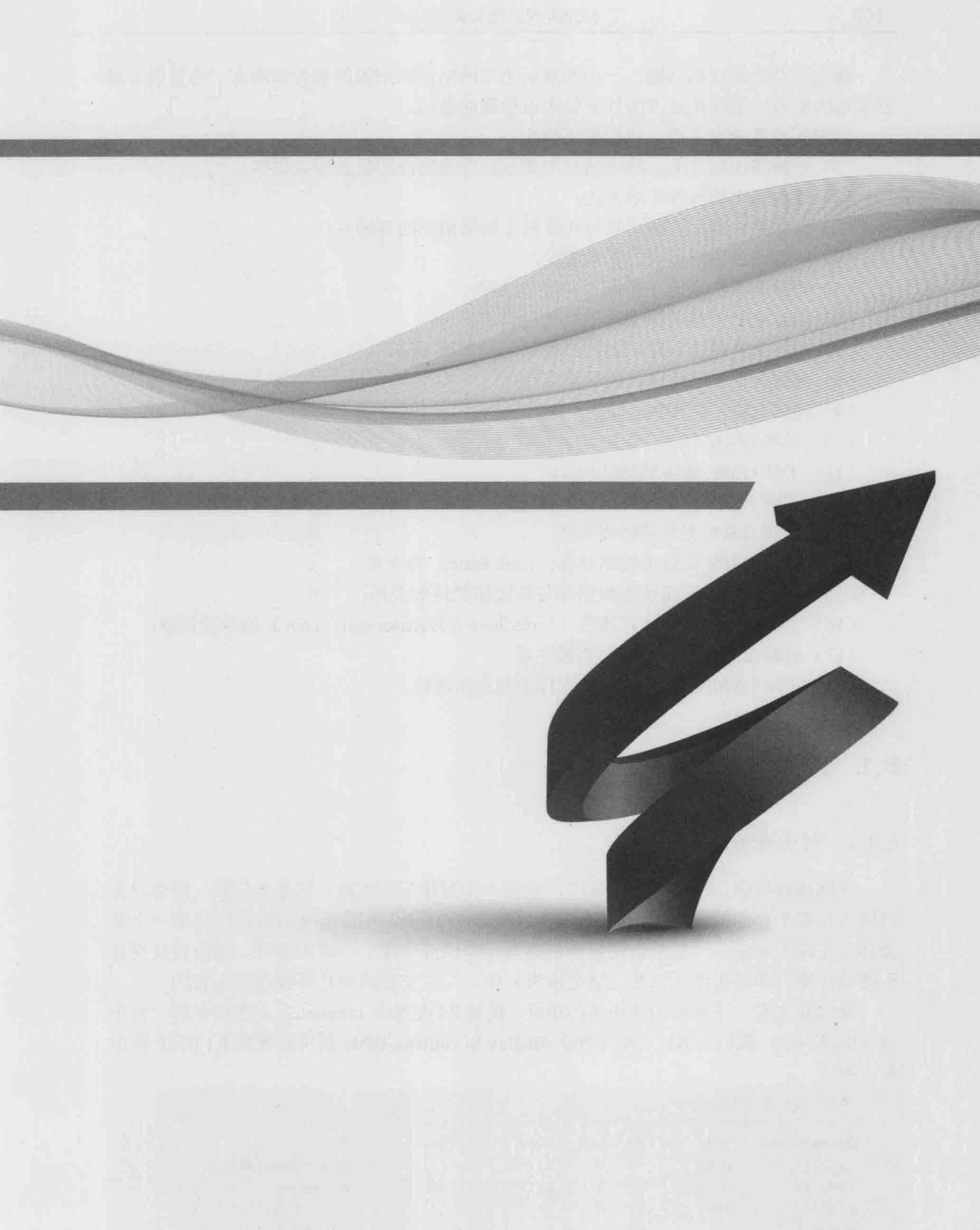
A．SYN　B．SYN+ACK　C．ACK

D．FIN　E．FIN+ACK

6．（单选）FTP 使用的端口号是？（　）

A．20　B．21　C．20 和 21

第8章
路由协议基础



路由及路由协议的问题，一向都被认为是网络技术知识的重点和难点。希望读者通过本章的学习，能够在这方面打下坚实而稳固的基础。

学习完本章内容之后，我们应该能够：

（1）理解路由的含义，路由的三个要素，以及路由的其他相关属性；
（2）理解路由的各种生成方式；
（3）理解路由器上的路由表与计算机上的路由表的异同点；
（4）了解路由协议分类的基本情况；
（5）理解 RIP 协议的工作原理和细节过程；
（6）熟悉 RIP 消息的报文结构及其封装情况；
（7）熟悉 RIP 路由环路问题的产生原因及解决方法；
（8）理解各种 RIP 定时器的功能和作用；
（9）理解 OSPF 与 RIP 之间的原理性区别；
（10）了解 OSPF 相比于 RIP 的各种优点；
（11）了解 OSPF 的区域化结构；
（12）了解 OSPF 协议报文的分类情况；
（13）了解 OSPF 支持的网络类型；
（14）理解 OSPF 协议中链路状态（Link State）的含义；
（15）理解 OSPF 链路状态数据库与最短路径树的关系；
（16）了解 OSPF 链路状态通告（Link-State Advertisement，LSA）的各种类型；
（17）理解 OSPF 邻居关系与邻接关系；
（18）了解 OSPF DR/BDR 的基本作用和选举过程。

8.1 路由的概念

8.1.1 什么是路由

在网络通信中，“路由（Route）”一词是一个网络层的术语，它是指从某一网络设备出发去往某个目的地的路径；而路由表（Routing Table）则是若干条路由信息的一个集合体。在路由表中，一条路由信息也被称为一个路由项或一个路由条目。路由表只存在于终端计算机和路由器（以及三层交换机）中，二层交换机中是不存在路由表的。

我们先来看一下实际的路由表的模样。假设 R1 是某个 internet 上正在运行的一台华为 AR 路由器，我们在 R1 上执行命令 **display ip routing-table** 便可查看到 R1 的 IP 路由表，如下。

```
<R1> display ip routing-table
---------------------------------------------------------------------------
Destination/Mask   Proto    Pre   Cost   Flags   NextHop     Interface
1.0.0.0/8          Direct   0     0      D       1.0.0.1     GigabitEthernet1/0/0
1.0.0.1/32         Direct   0     0      D       127.0.0.1   InLoopBack0
2.0.0.0/8          Static   60    0      D       12.0.0.2    GigabitEthernet1/0/1
2.1.0.0/16         RIP      100   1      D       12.0.0.2    GigabitEthernet1/0/1
```

```
12.0.0.0/30      Direct   0   0   D   12.0.0.1    GigabitEthernet1/0/1
12.0.0.1/32      Direct   0   0   D   127.0.0.1   InLoopBack0
……
```

在这个路由表中，每一行就是一条路由信息（一个路由项或一个路由条目）。通常情况下，一条路由信息由三个要素组成，它们分别是：目的地/掩码（Destination/Mask）、出接口（Interface）、下一跳 IP 地址（Next Hop）。我们现在以 Destination/Mask 为 2.0.0.0/8 这个路由项为例，来对路由信息的三个要素进行说明。

显然，2.0.0.0/8 是一个网络地址，掩码长度是 8。由于 R1 的 IP 路由表中存在 2.0.0.0/8 这个路由项，就说明 R1 知道自己所在的 internet 上存在一个网络地址为 2.0.0.0/8 的网络。需要特别说明的是，如果目的地/掩码中的掩码长度为 32，则目的地将是一个主机接口地址，否则目的地就是一个网络地址。通常，我们总是说一个路由项的目的地是一个网络地址（即目的网络地址），而把主机接口地址视为目的地的一种特殊情况。

从这个路由表中可以看到，2.0.0.0/8 这个路由项的出接口（Interface）是 GigabitEthernet1/0/1，其含义是：如果 R1 需要将一个 IP 报文送往 2.0.0.0/8 这个目的网络，那么 R1 应该把这个 IP 报文从 R1 的 GigabitEthernet1/0/1 接口发送出去。

从这个路由表中还可以看到，2.0.0.0/8 这个路由项的下一跳 IP 地址（Next Hop）是 12.0.0.2，其含义是：如果 R1 需要将一个 IP 报文送往 2.0.0.0/8 这个目的网络，则 R1 应该把这个 IP 报文从 R1 的 GigabitEthernet1/0/1 接口发送出去，并且这个 IP 报文离开 R1 的 GigabitEthernet1/0/1 接口后应该到达的下一个路由器的接口的 IP 地址是 12.0.0.2。需要指出的是，如果一个路由项的下一跳 IP 地址与出接口的 IP 地址相同，则说明出接口已经直连到了该路由项所指的目的网络（也就是说，出接口已经位于目的网络之中了）。还需要指出的是，下一跳 IP 地址所对应的那个主机接口与出接口一定是位于同一个二层网络（二层广播域）的。

总之，通常情况下，目的地/掩码（Destination/Mask）、出接口（Interface）、下一跳 IP 地址（Next Hop）是构成一个路由项的三个要素。然而，除了这三个要素外，一个路由项通常还包含其他一些属性，例如，产生这个路由项的 Protocol（路由表中 Proto 列），该路由项的 Preference（路由表中 Pre 列），该条路由的代价值（路由表中 Cost 列）等。

接下来我们解释一下路由器是如何进行 IP 路由表查询工作的。当路由器的 IP 转发模块接收到一个 IP 报文时，路由器将会根据这个 IP 报文的目的 IP 地址来进行 IP 路由表的查询工作，也就是将这个 IP 报文的目的 IP 地址与 IP 路由表的所有路由项逐项进行匹配。假设这个 IP 报文的目的 IP 地址为 x，路由器的某个路由项的目的地/掩码为 z/y，那么，如果 x 与 y 进行逐位“与”运算之后的结果等于 z，我们就说这个 IP 报文匹配上了 z/y 这个路由项；如果 x 与 y 进行逐位“与”运算之后的结果不等于 z，我们就说这个 IP 报文没有匹配上 z/y 这个路由项。

以前面的 IP 路由表为例，如果一个 IP 报文的目的 IP 地址为 2.1.0.1，那么这个 IP 报文就匹配上了 2.0.0.0/8 这个路由项，但是匹配不上 12.0.0.0/30 这个路由项。事实上，这个 IP 报文还可以匹配上 2.1.0.0/16 这个路由项。当一个 IP 报文同时匹配上了多个路由项时，路由器将根据“最长掩码匹配”原则来确定出一条最优路由，并根据最优路由来进行 IP 报文的转发。例如，目的地址为 2.1.0.1 的 IP 报文既能匹配上 2.0.0.0/8 这个路由

项，也能匹配上 2.1.0.0/16 这个路由项，但是后者的掩码长度大于前者的掩码长度，所以 2.1.0.0/16 这条路由就被确定为目的地址为 2.1.0.1 的 IP 报文的最优路由。路由器总是根据最优路由来进行 IP 报文的转发的。

计算机也会进行 IP 路由表的查询工作。当计算机的网络层封装好了等待发送的 IP 报文后，就会根据 IP 报文的目的 IP 地址去查询自己的 IP 路由表。计算机上 IP 路由表的查询过程与路由器上 IP 路由表的查询过程完全一样（例如，同样要遵循最长掩码匹配原则等），这里不再赘述。最后，计算机将根据查表而确定出的最优路由将相应的 IP 报文发送出去。

8.1.2 路由信息的来源

我们知道，一个 IP 路由表中包含了若干条路由信息。那么，这些路由信息是从何而来的呢？或者说，这些路由信息是如何生成的呢？

路由信息的生成方式总共有三种：设备自动发现、手工配置、通过动态路由协议生成。我们把设备自动发现的路由信息称为直连路由（Direct Route），把手工配置的路由信息称为静态路由（Static Route），把网络设备通过运行动态路由协议而得到路由信息称为动态路由（Dynamic Route）。上一小节中所展示的 R1 的 IP 路由表中，Protocol 一列为 Direct 的那些路由项就是 R1 自动发现的直连路由信息，Protocol 一列为 Static 的那些路由项就是人工配置的静态路由信息，Protocol 一列为 RIP 的那些路由项就是 R1 通过运行 RIP 路由协议而得到的动态路由信息。

1. 直连路由

网络设备启动之后，当设备接口的状态为 UP 时，设备就能够自动发现去往与自己的接口直接相连的网络的路由。当我们说某一网络是与某台网络设备的某个接口直接相连（直连）的时候，是指这台设备的这个接口已经位于这个网络之中了，而这里所说的某一网络是指某个二层网络（二层广播域）。当我们说某一网络是与某台网络设备直接相连（直连）的时候，是指这个网络是与这个设备的某个接口直接相连的。

如图 8-1 所示，路由器 R1 的 GE1/0/0 接口的状态为 UP 时，R1 便可以根据 GE1/0/0 接口的 IP 地址 1.0.0.1/24 推断出 GE1/0/0 接口所在的网络的网络地址为 1.0.0.0/24。于是，R1 便会将 1.0.0.0/24 作为一个路由项填写进自己的路由表，这条路由的目的地/掩码为 1.0.0.0/24，出接口为 GE1/0/0，下一跳 IP 地址是与出接口的 IP 地址相同的，即 1.0.0.1。由于这条路由是直连路由，所以其 Protocol 属性为 Direct。另外，对于直连路由，其 Cost 的值总是为 0。

类似地，路由器 R1 还会自动发现另外一条直连路由，该路由的目的地/掩码为 2.0.0.0/24，出接口为 GE2/0/0，下一跳 IP 地址是 2.0.0.1，Protocol 属性为 Direct，Cost 的值为 0。

同样，PC 1 也会自动发现一条直连路由，该路由的目的地/掩码为 1.0.0.0/24，出接口为 PC 1 的网口（假设 PC 1 只有一个网口），下一跳 IP 地址是 1.0.0.2，Protocol 属性为 Direct，Cost 的值为 0。

最后，PC 2 也会自动发现一条去往 2.0.0.0/24 的直连路由，这里不再赘述。

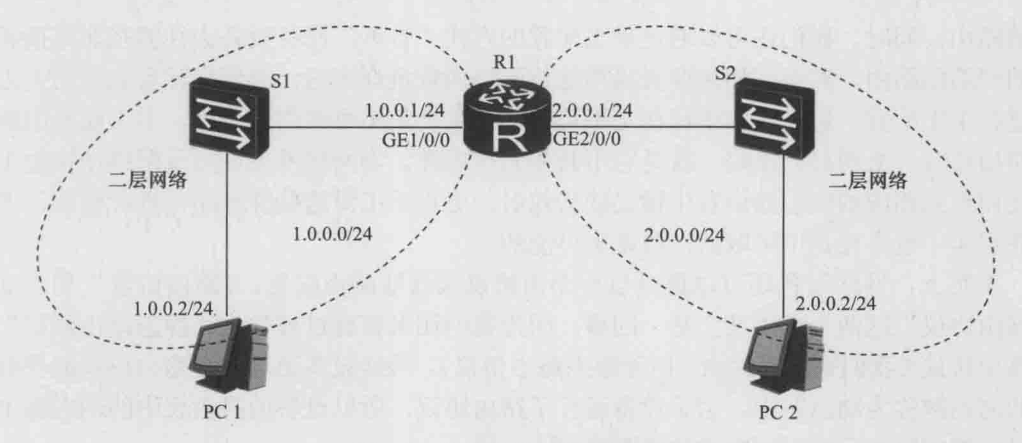


图 8-1 设备自动发现直连路由

2. 静态路由

如图 8-2 所示，R1 显然是可以自动发现 1.0.0.0/8 和 12.0.0.0/30 这两条直连路由的。然而，R1 无法自动发现 2.0.0.0/8 这条路由。为此，我们可以人为地在 R1 上手工配置一条路由，该路由的目的地/掩码为 2.0.0.0/8，出接口为 R1 的 GE1/0/1，下一跳 IP 地址为 R2 的 GE1/0/1 接口的 IP 地址 12.0.0.2，Cost 的值可以人为地设定为 0（也可以是其他我们希望的值）。这条路由出现在 R1 的路由表中时，Protocol 属性将会是 Static，表示是一条静态路由。

当然，我们也可以在 R2 上手工配置一条去往 1.0.0.0/8 的静态路由，出接口为 R2 的 GE1/0/1，下一跳 IP 地址为 R1 的 GE1/0/1 接口的 IP 地址 12.0.0.1，Cost 的值可以人为地设定为 0（也可以是其他我们希望的值）。

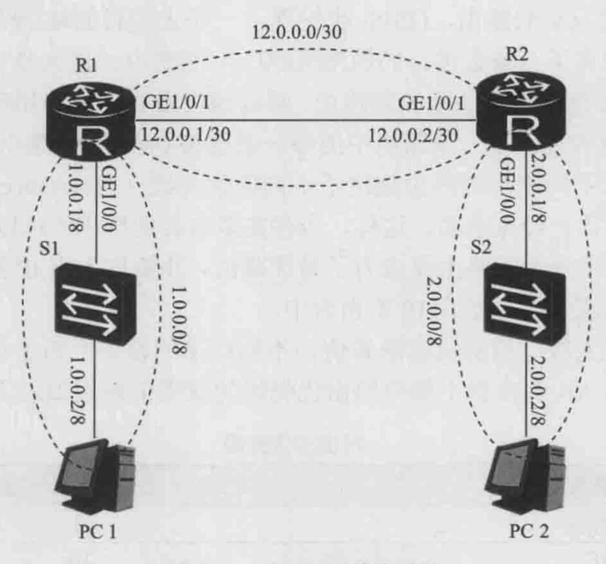


图 8-2 手工配置静态路由

3. 动态路由

前面介绍了直连路由和静态路由。网络设备可以自动发现去往与自己直接相连的网

络的路由，同时，我们还可以通过手工配置的方式“告诉”网络设备去往哪些非直接相连的网络的路由。然而，如果非直接相连的网络的数量众多时，必然会耗费大量的人力来进行手工配置，这在现实中往往是不可取的，甚至是不可能的。另外，手工配置的静态路由还有一个明显的缺陷，就是它不具备自适应性。当网络发生故障或网络结构发生改变而导致相应的静态路由发生错误或失效时，必须手工对这些静态路由进行修改，而这在现实中也往往是不可取的，或是不可能的。

事实上，网络设备还可以通过运行路由协议来获取路由信息。“路由协议”和“动态路由协议”这两个术语其实是一回事，因为我们还未曾有过被称为“静态路由协议”的路由协议（我们有静态路由，但无静态路由协议）。网络设备通过运行路由协议而获取到的路由被称为动态路由。由于设备运行了路由协议，所以设备的路由表中的动态路由信息能够实时地反映出网络结构的变化。

需要特别指出的是，一台路由器是可以同时运行多种路由协议的。比如，一台路由器可以同时运行 RIP 路由协议和 OSPF 路由协议。此时，该路由器除了会创建并维护一个 IP 路由表外，还会分别创建并维护一个 RIP 路由表和一个 OSPF 路由表。RIP 路由表用来专门存放 RIP 协议发现的所有路由，OSPF 路由表用来专门存放 OSPF 协议发现的所有路由。通过一些优选法则的筛选后，某些 RIP 路由表中的路由项以及某些 OSPF 路由表中的路由项才能被加入进 IP 路由表，而路由器最终是根据 IP 路由表来进行 IP 报文的转发工作的。

同时需要提请读者注意的是，计算机是不运行任何路由协议的。计算机上只有一个 IP 路由表。

8.1.3 路由的优先级

假设一台华为 AR 路由器同时运行了 RIP 和 OSPF 这两种路由协议，RIP 发现了一条去往目的地/掩码为 z/y 的路由，OSPF 也发现了一条去往目的地/掩码为 z/y 的路由。另外，我们还手工配置了一条去往目的地/掩码为 z/y 的路由。也就是说，该设备同时获取了去往同一目的地/掩码的三条不同的路由，那么该设备究竟会采用哪一条路由来进行 IP 报文的转发呢？或者说，这三条路由中的哪一条会被加入进 IP 路由表呢？

事实上，我们给不同来源的路由规定了不同的优先级（Preference），并规定优先级的值越小，则路由的优先级就越高。这样，当存在多条目的地/掩码相同，但来源不同的路由时，则具有最高优先级的路由便成为了最优路由，并被加入进 IP 路由表中，而其他路由则处于未激活状态，不显示在 IP 路由表中。

设备上的路由优先级一般都具有缺省值。不同厂家的设备上对于优先级的缺省值的规定可能不同。华为 AR 路由器上部分路由优先级的缺省值规定如表 8-1 所示。

表 8-1 路由的优先级

路由来源	优先级的缺省值
直连路由	0
OSPF	10
静态路由	60
RIP	100
BGP	255

回头看看本小节一开始提出的问题，相信读者的心中已经有了正确的答案。

8.1.4　路由的开销

路由的开销（Cost）是路由的一个非常重要的属性。一条路由的开销是指到达这条路由的目的地/掩码需要付出的代价值。同一种路由协议发现有多条路由可以到达同一目的地/掩码时，将优选开销最小的路由，即只把开销最小的路由加入进本协议的路由表中。

不同的路由协议对于开销的具体定义是不同的。比如，RIP 协议只能将“跳数（Hop Count）”作为开销。所谓跳数，就是指到达目的地/掩码需要经过的路由器的个数。如图 8-3 所示，假设路由器 R1、R2、R3 均运行 RIP 路由协议。通过运行 RIP 协议，R1 会发现两条去往 2.0.0.0/8 的路由，第一条路由的出接口是 R1 的 GE1/0/0 接口，下一跳 IP 地址是 R2 的 GE1/0/0 接口的 IP 地址，开销（跳数）为 3（因为根据这条路由从 R1 去往 2.0.0.0/8 需要经过 R1、R2、R3 这 3 个路由器）；第二条路由的出接口是 R1 的 GE2/0/0 接口，下一跳 IP 地址是 R3 的 GE1/0/0 接口的 IP 地址，开销（跳数）为 2（因为根据这条路由从 R1 去往 2.0.0.0/8 只需要经过 R1 和 R3 这两个路由器）。显然，第二条路由的开销小于第一条路由的开销，所以第二条路由为最优路由，并将被加入进 R1 的 RIP 路由表。

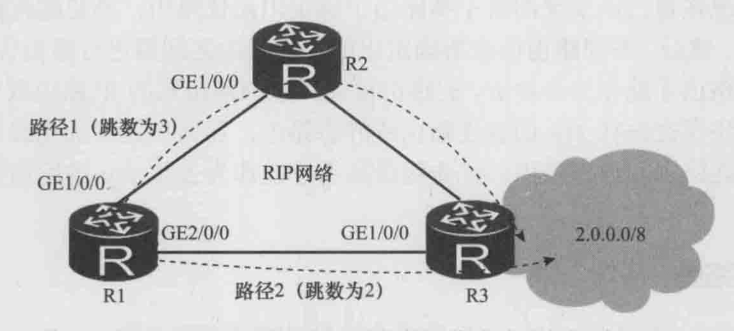


图 8-3　RIP 协议只能以“跳数”作为路由的开销

同一种路由协议发现有多条路由可以到达同一目的地/掩码时，并且这些路由的开销又是相等的，那该怎么办呢？如图 8-4 所示，假设路由器 R1、R2、R3、R4 均运行 RIP 路由协议。通过运行 RIP 协议，R1 会发现两条去往 2.0.0.0/8 的路由，第一条路由的出接口是 R1 的 GE1/0/0 接口，下一跳 IP 地址是 R2 的 GE1/0/0 接口的 IP 地址，开销为 3（因为根据这条路由从 R1 去往 2.0.0.0/8 需要经过 R1、R2、R3 这 3 个路由器）；第二条路由的出接口是 R1 的 GE2/0/0 接口，下一跳 IP 地址是 R4 的 GE1/0/0 接口的 IP 地址，开销为 3（因为根据这条路由从 R1 去往 2.0.0.0/8 需要经过路由器 R1、R4、R3 这 3 个路由器）。由于这两条路由的代价（开销）是相等的，所以它们被称为等价路由。在这种情况下，这两条路由都会被加入进 R1 的 RIP 路由表。如果 RIP 路由表中的这两条路由能够被优选进入 IP 路由表的话，那么 R1 在转发去往 2.0.0.0/8 的流量时，一部分流量会根据第一条路由来进行转发，另一部分流量会根据第二条路由来进行转发，这种情况也被称为负载分担（Load Balance）。

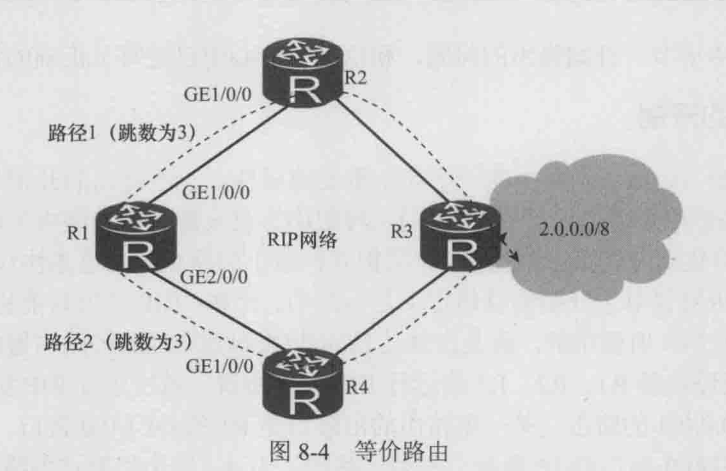


图 8-4 等价路由

需要特别强调的是，不同的路由协议对于开销的具体定义是不同的，开销值大小的比较只在同一种路由协议内才有意义，不同路由协议之间的路由开销值没有可比性，也不存在换算关系。

如果一台路由器同时运行了多种路由协议，并且对于同一目的地/掩码（假设为 z/y），每一种路由协议都发现了一条或多条路由，在这种情况下，每一种路由协议都会根据开销值的比较情况在自己所发现的若干条路由中确定出最优路由，并将最优路由放进本协议的路由表中。然后，不同路由协议所确定出的最优路由之间再进行路由优先级的比较，优先级最高的路由才能作为去往 z/y 的路由被加入进该路由器的 IP 路由表中。注意，如果该路由器上还存在去往 z/y 的直连路由或静态路由，那么在进行优先级比较的时候也要考虑这些直连路由和静态路由，优先级最高者才能作为去往 z/y 的路由被最终加入进 IP 路由表中。

8.1.5 默认路由

我们把目的地/掩码为 0.0.0.0/0 的路由称为默认路由或缺省路由（Default Route）。如果默认路由是由路由协议产生的，则称为动态默认路由；如果默认路由是由手工配置而成的，则称为静态默认路由。默认路由是一种非常特殊的路由，因为任何一个待发送或待转发的 IP 报文都是可以和默认路由匹配上的，虽然掩码匹配长度为 0。

计算机或路由器的 IP 路由表中可能存在默认路由，也可能不存在默认路由。如果网络设备的 IP 路由表中存在默认路由，那么当一个待发送或待转发的 IP 报文不能匹配 IP 路由表中的任何非默认路由时，就会根据默认路由来进行发送或转发；如果网络设备的 IP 路由表中不存在默认路由，那么当一个待发送或待转发的 IP 报文不能匹配 IP 路由表中的任何路由时，该 IP 报文就会被直接丢弃。

8.1.6 计算机上的路由表与路由器上的路由表

计算机上的 IP 路由表的规模一般都很小，通常只包含一、二十条路由。对于路由器来说，其 IP 路由表的规模大小变化很大，并且是与该路由器所运行的路由协议及该路由器在整个网络中的位置紧密相关的。路由器上的 IP 路由表可能包含几条、几十条、几百条、几千条、几万条、几十万条，甚至上百万条路由。

计算机是不运行任何路由协议的，所以计算机的 IP 路由表中的路由要么是直连路由，要么是手工配置的静态路由，还有就是计算机的操作系统代替我们的手工配置而配置出来的各种路由。路由器的 IP 路由表中的路由可以有直连路由，可以有静态路由，但更多的都是通过运行路由协议而获得的动态路由。路由器上除了存在 IP 路由表外，还存在为每个运行的路由协议专门创建并维护的路由表。

8.1.7　静态路由配置示例

对于图 8-5 所示的简单的 internet，可以通过在 R1 和 R2 上配置静态路由来实现各个 PC 之间的互通。

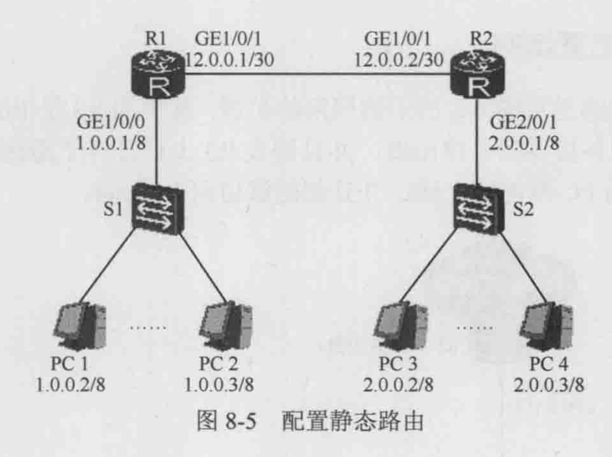


图 8-5　配置静态路由

1．配置思路

在路由器 R1 上配置一条静态路由，目的地/掩码为 2.0.0.0/8，下一跳 IP 地址为 R2 的 GE1/0/1 接口的 IP 地址 12.0.0.2，出接口为 R1 的 GE1/0/1 接口。

在路由器 R2 上配置一条静态路由，目的地/掩码为 1.0.0.0/8，下一跳 IP 地址为 R1 的 GE1/0/1 接口的 IP 地址 12.0.0.1，出接口为 R2 的 GE1/0/1 接口。

2．配置步骤

在路由器上配置静态路由时，需要进入到系统视图，然后执行命令 **ip route-static** *ip-address* { *mask* | *mask-length* } { *nexthop-address* | *interface-type interface-number* [*nexthop-address*] } [**preference** *preference*]。其中，*ip-address* { *mask* | *mask-length* }表示目的地/掩码，*nexthop-address* 表示下一跳 IP 地址，*interface-type interface-number* 表示出接口，*preference* 表示路由的优先级。

#配置 R1。

```
<R1> system-view
[R1] ip route-static 2.0.0.0 8 12.0.0.2 gigabitethernet 1/0/1
```

#配置 R2。

```
<R2> system-view
[R2] ip route-static 1.0.0.0 8 12.0.0.1 gigabitethernet 1/0/1
```

配置完成后，我们通常需要对配置好的路由进行确认。以 R1 为例，使用 **display ip routing-table** 命令查看其 IP 路由表。

```
<R1> display ip routing-table
```

```
Route Flags: R - relay, D - download to fib
------------------------------------------------------------------------------
Routing Tables: Public
         Destinations : 5          Routes : 5
Destination/Mask    Proto     Pre   Cost    Flags    NextHop       Interface
1.0.0.0/8           Direct    0     0       D        1.0.0.1       GigabitEthernet1/0/0
1.0.0.1/32          Direct    0     0       D        127.0.0.1     InLoopBack0
2.0.0.0/8           Static    60    0       D        12.0.0.2      GigabitEthernet1/0/1
12.0.0.0/30         Direct    0     0       D        12.0.0.1      GigabitEthernet1/0/1
12.0.0.1/32         Direct    0     0       D        127.0.0.1     InLoopBack0
```

从回显信息中我们可以看到，R1 的 IP 路由表中已经有了一条关于 2.0.0.0/8 的静态路由。注意，静态路由的默认优先级的值为 60。

8.1.8 默认路由配置示例

图 8-6 所示的网络是对图 4-5 所示的网络的扩展，其中的 R3 是 ISP（Internet Service Provider，Internet 服务提供商）路由器，并且假设 R3 上已经有了通往 Internet 的路由。网络需求是：所有的 PC 都能够互通，并且都能够访问 Internet。

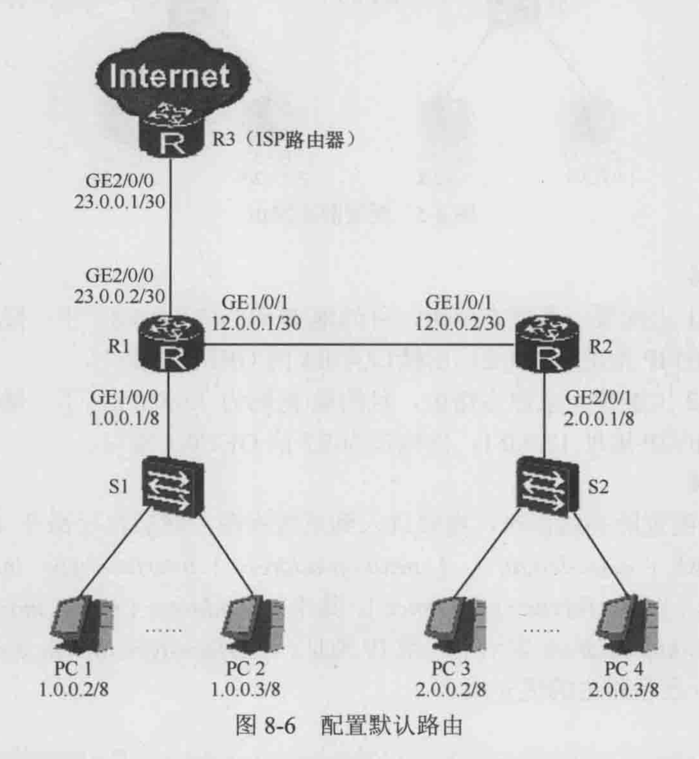


图 8-6 配置默认路由

1. 配置思路

在路由器 R1 上配置一条静态路由，目的地/掩码为 2.0.0.0/8，下一跳 IP 地址为 R2 的 GE1/0/1 接口的 IP 地址 12.0.0.2，出接口为 R1 的 GE1/0/1 接口。另外，在 R1 上配置一条默认路由，该默认路由的下一跳 IP 地址为 R3 的 GE2/0/0 接口的 IP 地址 23.0.0.1，出接口为 R1 的 GE2/0/0 接口。

在路由器 R2 上配置一条静态路由，目的地/掩码为 1.0.0.0/8，下一跳 IP 地址为 R1

的 GE1/0/1 接口的 IP 地址 12.0.0.1，出接口为 R2 的 GE1/0/1 接口。另外，在 R2 上配置一条默认路由，该默认路由的下一跳 IP 地址为 R1 的 GE1/0/1 接口的 IP 地址 12.0.0.1，出接口为 R2 的 GE1/0/1 接口。

在 R3 上配置分别配置一条去往 1.0.0.0/8 和 2.0.0.0/8 的静态路由，下一跳 IP 地址均为 R1 的 GE2/0/0 接口的 IP 地址 23.0.0.2，出接口均为 R3 的 GE2/0/0 接口。

2. 配置步骤

#配置 R1。

```
<R1> system-view
[R1] ip route-static 2.0.0.0 8 12.0.0.2 gigabitethernet 1/0/1
[R1] ip route-static 0.0.0.0 0 23.0.0.1 gigabitethernet 2/0/0
```

#配置 R2。

```
<R2> system-view
[R2] ip route-static 1.0.0.0 8 12.0.0.1 gigabitethernet 1/0/1
[R2] ip route-static 0.0.0.0 0 12.0.0.1 gigabitethernet 1/0/1
```

#配置 R3。

```
<R3> system-view
[R3] ip route-static 1.0.0.0 8 23.0.0.2 gigabitethernet 2/0/0
[R3] ip route-static 2.0.0.0 8 23.0.0.2 gigabitethernet 2/0/0
```

配置完成后，我们通常需要对配置好的路由进行确认。以 R1 为例，使用 **display ip routing-table** 命令查看其 IP 路由表。

```
<R1> display  ip  routing-table
Route Flags: R – relay, D - download to fib
------------------------------------------------------------------------------
Routing Tables: Public
         Destinations : 8          Routes : 8
Destination/Mask   Proto    Pre   Cost   Flags   NextHop     Interface
0.0.0.0/0          Static   60    0      RD      23.0.0.1    GigabitEthernet2/0/0
1.0.0.0/8          Direct   0     0      D       1.0.0.1     GigabitEthernet1/0/0
1.0.0.1/32         Direct   0     0      D       127.0.0.1   InLoopBack0
2.0.0.0/8          Static   60    0      D       12.0.0.2    GigabitEthernet1/0/1
12.0.0.0/30        Direct   0     0      D       12.0.0.1    GigabitEthernet1/0/1
12.0.0.1/32        Direct   0     0      D       127.0.0.1   InLoopBack0
23.0.0.0/30        Direct   0     0      D       23.0.0.2    GigabitEthernet2/0/0
23.0.0.2/32        Direct   0     0      D       127.0.0.1   InLoopBack0
```

从回显信息中我们可以看到，R1 的路由表中已经有了一条静态默认路由。

8.1.9　练习题

1.（单选）以下 4 条路由都以静态路由的形式存在于某路由器的 IP 路由表中，那么该路由器对于目的 IP 地址为 8.1.1.1 的 IP 报文将根据哪条路由来进行转发？（　）

A．0.0.0.0/0　　B．8.0.0.0/8　　C．8.1.0.0/16　　D．18.0.0.0/16

2.（多选）路由信息的来源有哪些？（　）

A．设备自动发现的直连路由　　B．手工配置的静态路由

C．路由协议发现的路由　　D．以上都不是

3.（单选）某路由器同时运行了 RIP 和 OSPF 两种路由协议，该路由器自动发现了一条去往 9.0.0.0/8 的直连路由，RIP 发现了一条去往 9.0.0.0/8 的路由，OSPF 发现了一条去往 9.0.0.0/8 的路由，另外还手工配置了一条去往 9.0.0.0/8 路由。那么，默认情况下，

哪一条路由才会被加入到该路由器的 IP 路由表中？（ ）

A．路由器自动发现的那条去往 9.0.0.0/8 的直连路由

B．手工配置的那条去往 9.0.0.0/8 的静态路由

C．RIP 发现的那条去往 9.0.0.0/8 的路由

D．OSPF 发现的那条去往 9.0.0.0/8 的路由

4.（单选）静态路由的默认优先级为？（ ）

A．0　　B．60　　C．100　　D．120

5.（单选）某路由器同时运行了 RIP 和 OSPF 两种路由协议，RIP 发现了两条去往 9.0.0.0/8 的路由，其中一条路由的 Cost 为 5，另一条路由的 Cost 为 7；OSPF 发现了一条去往 9.0.0.0/8 的路由，Cost 为 100；另外还手工配置了一条去往 9.0.0.0/8 的静态路由，Cost 为 60。那么，默认情况下，哪一条路由才会被加入到该路由器的 IP 路由表中？（ ）

A．9.0.0.0/8（Static，Cost 为 60）　　B．9.0.0.0/8（RIP，Cost 为 5）

C．9.0.0.0/8（RIP，Cost 为 7）　　D．9.0.0.0/8（OSPF，Cost 为 100）

6.（单选）以下 4 条路由都以静态路由的形式存在于某路由器的 IP 路由表中，那么该路由器对于目的 IP 地址为 8.1.1.1 的 IP 报文将根据哪条路由来进行转发？（ ）

A．0.0.0.0/0　　B．8.2.0.0/16　　C．8.1.2.0/24　　D．18.1.0.0/16

8.2 RIP 协议

8.2.1 路由协议概述

首先，我们简单解释一下自治系统（Autonomous System，AS）的概念。在网络通信中，一个自治系统是指由若干个二层网络及若干台路由器组成的集合，集合中的这些网络及路由器均属于同一个管理机构。由于规模大小的不同，一个 internet 可能只包含一个自治系统，也可能包含多个不同的自治系统。例如，图 8-7 所示的 internet 便包含了 3 个不同的自治系统，分别是自治系统 X、自治系统 Y、自治系统 Z。

路由协议分为两大类，一类称为 IGP（Interior Gateway Protocol，内部网关协议），另一类称为 EGP（Exterior Gateway Protocol，外部网关协议）。IGP 的成员有 RIP（Routing Information Protocol）协议、OSPF（Open Shortest Path First）协议、IS-IS（Intermediate System to Intermediate System）协议等。EGP 虽然也有若干个成员协议，但目前在实际的网络中得到应用的协议只有一个，那就是 BGP（Border Gateway Protocol）协议。

通常情况下，一个自治系统中的所有路由器需要运行同一种具体的、由该自治系统的管理机构指定的 IGP 协议（有的情况下还可能会同时运行不同的 IGP 协议）。IGP 协议的运行，将会使得自治系统中的每一台路由器都能够发现通往本自治系统内各个目的网络的路由。在一个由多个自治系统组成的 internet 中，通常还需要通过 BGP 协议来实现不同自治系统之间的路由交换。通过这种交换，一台路由器不仅能发现通往本自治系统内各个目的网络的路由，而且还能够发现通往其他自治系统中的目的网络

的路由。

例如，在图 8-7 所示的 internet 中，自治系统 X 中的所有路由器（包括路由器 Rx）均运行 RIP 协议，自治系统 Y 中的所有路由器（包括路由器 Ry）均运行 OSPF 协议，自治系统 Z 中的所有路由器（包括路由器 Rz）均运行 OSPF 协议。同时，Rx、Ry、Rz 还要运行 BGP 协议，以实现不同自治系统之间的路由交换。通过这样的路由架构，每一台路由器便可以发现通往该 internet 中任何目的网络的路由。

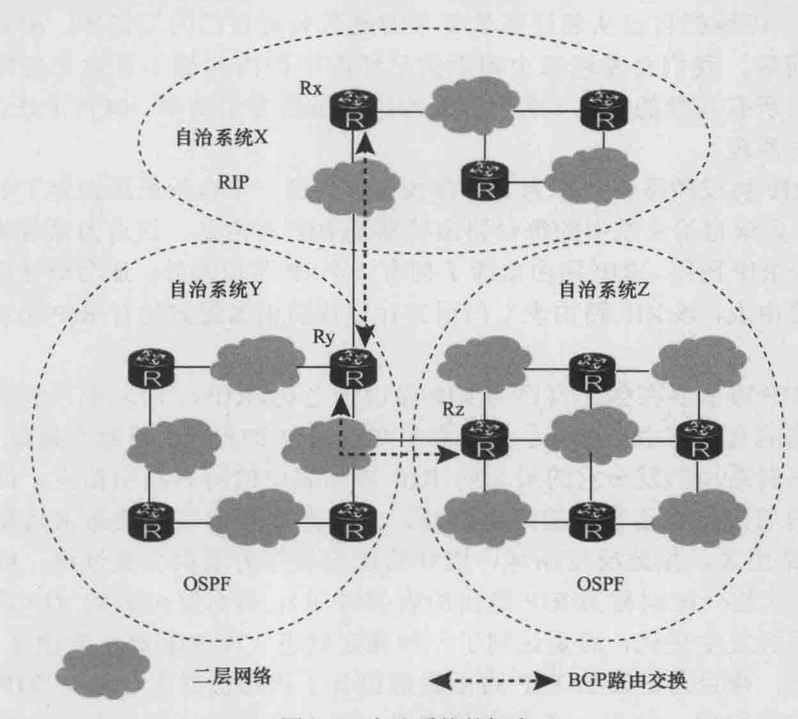


图 8-7　自治系统的概念

8.2.2　RIP 协议的基本原理

RIP（Routing Information Protocol，路由信息协议）是一种基于距离矢量（Distance Vector，简称 DV）算法的 IGP 协议，其协议优先级的值为 100。相比于其他各种路由协议，RIP 协议最为简单且易于实现。

RIP 协议只能以“跳数”来定义路由的开销。所谓跳数，就是指到达目的地需要经过的路由器的个数。例如，在图 8-8 所示的网络中，路由器 R1 去往网络 A、网络 B、网络 C、网络 D 的跳数分别为 1、2、3、4。RIP 协议规定，跳数等于或大于 16 的路由将被视为不可达的路由，这一限制使得 RIP 协议一般只应用于规模较小的网络。

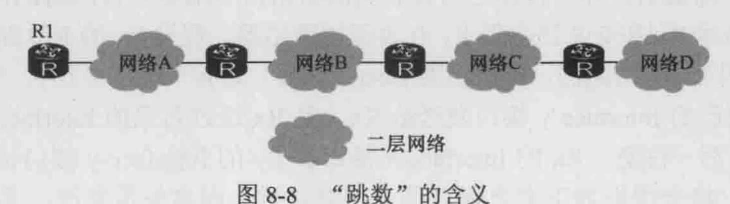


图 8-8　“跳数”的含义

在描述 RIP 协议的基本原理之前，我们先来看一个带有比喻性的游戏活动。假设在一个教室里坐满了新同学，坐中间的每个同学有前、后、左、右 4 个邻居，坐边上的同学有 3 个邻居，坐角落的同学有 2 个邻居。游戏开始前，假定每个同学都只知道自己的所有邻居的姓名，也就是说，每个同学的“记忆库”中只有自己的几个邻居的姓名。游戏开始后，每个同学都周期性地把自己最新的记忆库中的所有姓名悄悄地告诉给自己的所有邻居（每个同学只能听见邻居对自己说的话），同时不停地把自己从邻居那里听来的姓名装进自己的记忆库。游戏持续了足够长的时间后，我们会发现每个同学的记忆库中的内容都不再发生变化，并且都包含了全班所有同学的姓名。这个游戏的过程虽然非常简单，但它正好体现了 RIP 协议的基本原理。

运行 RIP 协议的路由器称为 RIP 路由器。假设一个自治系统选定了 RIP 协议作为其 IGP，则该自治系统中的每台路由器都是 RIP 路由器，该自治系统本身也通常被称为一个 RIP 网络。RIP 路由器除了拥有一个 IP 路由表外，还会单独创建并维护一个 RIP 路由表，该 RIP 路由表专门用来存放该路由器通过运行 RIP 协议而发现的路由。

一台 RIP 路由器在创建自己的 RIP 路由表之初，RIP 路由表中只包含了该路由器自动发现的直连路由。在一个 RIP 网络中，每台 RIP 路由器都会每隔 30 秒钟向它所有的邻居路由器发布它的最新的 RIP 路由表中的所有路由信息，同时又不断地接收它的邻居路由器发来的路由信息，并根据这些接收到的路由信息来更新自己的 RIP 路由表，如此反复循坏，这样的过程被称为路由交换过程。经过足够长的时间之后（这一时间称为 RIP 路由的收敛时间），每台路由器的 RIP 路由表中的路由信息不再发生变化，而是达到了一种稳定状态（即 RIP 路由实现了收敛）。在稳定状态下，每台路由器的 RIP 路由表都包含了该路由器去往整个 RIP 网络中各个目的网络的路由。注意，在稳定状态下，路由交换过程仍会继续进行。当网络的结构发生改变后，稳定状态会被打破，但随着路由交换过程的继续进行，经过足够长的时间之后，每台路由器的 RIP 路由表又会达到新的稳定状态（即 RIP 路由重新实现了收敛）。

8.2.3 RIP 路由表的形成

一台路由器在创建自己的 RIP 路由表之初，RIP 路由表中只包含了该路由器自动发现的直连路由。随后，该路由器不断地接收它的邻居路由器发来的路由信息，并根据这些接收到的路由信息来更新自己的 RIP 路由表。同时，该路由器每隔 30 秒钟会向它所有的邻居路由器发布它的最新的 RIP 路由表中的所有路由信息。

那么，RIP 路由器是如何根据它所接收到的路由信息来更新自己的 RIP 路由表的呢？假设 RIP 路由器 Rx 和 RIP 路由器 Ry 互为邻居路由器。假设 Ry 的 RIP 路由表中存在一条目的地/掩码为 z/y 的路由，该路由的 Cost（跳数）为 N（$1 \leqslant N \leqslant 16$）。当 Ry 把这条路由信息通过自己的 Interface-y 接口发送给 Rx，且 Rx 通过自己的 Interface-x 接口接收到这条路由信息后（注意，Rx 的 Interface-x 接口和 Ry 的 Interface-y 接口位于同一个二层网络中），Rx 将会根据如下的更新算法（该算法称为距离矢量算法，其基本思想是由

Bellman-Ford 提出的，所以也称为 Bellman-Ford 算法）来更新自己的 RIP 路由表。在下面的算法中，如果 N+1 小于 16，则规定 M 的值为 N+1；如果 N+1 等于或大于 16，则规定 M 的值为 16。

（1）如果 Rx 的 RIP 路由表中不存在目的地/掩码为 z/y 的路由项，则 Rx 会在自己的 RIP 路由表中添加一个路由项，该路由项的目的地/掩码为 z/y，出接口为 Rx 的 Interface-x 接口，下一跳 IP 地址为 Ry 的 Interface-y 接口的 IP 地址，Cost 为 M。

（2）如果 Rx 的 RIP 路由表中已经存在一条目的地/掩码为 z/y 的路由项，且该路由项的下一跳 IP 地址为 Ry 的 Interface-y 接口的 IP 地址，则 Rx 会将该路由项的出接口更新为 Rx 的 Interface-x 接口（其实，更新后的出接口与更新前的出接口均为 Rx 的 Interface-x 接口），下一跳 IP 地址更新为 Ry 的 Interface-y 接口的 IP 地址（其实，更新后的下一跳 IP 地址与更新前的下一跳 IP 地址均为 Ry 的 Interface-y 接口的 IP 地址），Cost 更新为 M。

（3）如果 Rx 的 RIP 路由表中已经存在一条目的地/掩码为 z/y 的路由项，且该路由项的下一跳 IP 地址不是 Ry 的 Interface-y 接口的 IP 地址，并且该路由项的 Cost 大于 M，则 Rx 会将该路由项的出接口更新为 Rx 的 Interface-x 接口，下一跳 IP 地址更新为 Ry 的 Interface-y 接口的 IP 地址，Cost 更新为 M。

（4）如果 Rx 的 RIP 路由表中已经存在一条目的地/掩码为 z/y 的路由项，且该路由项的下一跳 IP 地址不是 Ry 的 Interface-y 接口的 IP 地址，并且该路由项的 Cost 小于或等于 M，则该路由项的出接口、下一跳 IP 地址以及 Cost 均保持不变。也就是说，Rx 不会对该路由项进行更新。



R2的RIP路由表（初态）

目的地/掩码	出接口	下一跳IP地址	Cost
2.0.0.0/8	GE1/0/0	2.0.0.1	1
12.0.0.0/8	GE2/0/0	12.0.0.2	1
23.0.0.0/8	GE3/0/0	23.0.0.1	1

R1的RIP路由表（初态）

目的地/掩码	出接口	下一跳IP地址	Cost
1.0.0.0/8	GE1/0/0	1.0.0.1	1
12.0.0.0/8	GE2/0/0	12.0.0.1	1

R3的RIP路由表（初态）

目的地/掩码	出接口	下一跳IP地址	Cost
3.0.0.0/8	GE1/0/0	3.0.0.1	1
23.0.0.0/8	GE2/0/0	23.0.0.2	1

图 8-9　初始状态下的 RIP 路由表

对于 RIP 路由表的形成过程，图 8-9 和图 8-10 给出了一个示例。建议读者朋友们根据这个例子亲自推演一下从初始的 RIP 路由表到稳定的 RIP 路由表的全过程。



R2的RIP路由表（稳态）

目的地/掩码	出接口	下一跳IP地址	Cost
2.0.0.0/8	GE1/0/0	2.0.0.1	1
12.0.0.0/8	GE2/0/0	12.0.0.2	1
23.0.0.0/8	GE3/0/0	23.0.0.1	1
1.0.0.0/8	GE2/0/0	12.0.0.1	2
3.0.0.0/8	GE3/0/0	23.0.0.2	2

R1的RIP路由表（稳态）

目的地/掩码	出接口	下一跳IP地址	Cost
1.0.0.0/8	GE1/0/0	1.0.0.1	1
12.0.0.0/8	GE2/0/0	12.0.0.1	1
2.0.0.0/8	GE2/0/0	12.0.0.2	2
23.0.0.0/8	GE2/0/0	12.0.0.2	2
3.0.0.0/8	GE2/0/0	12.0.0.2	3

R3的RIP路由表（稳态）

目的地/掩码	出接口	下一跳IP地址	Cost
3.0.0.0/8	GE1/0/0	3.0.0.1	1
23.0.0.0/8	GE2/0/0	23.0.0.2	1
2.0.0.0/8	GE2/0/0	23.0.0.1	2
12.0.0.0/8	GE2/0/0	23.0.0.1	2
1.0.0.0/8	GE2/0/0	23.0.0.1	3

图 8-10　稳定状态下的 RIP 路由表

8.2.4　RIP 消息的格式

RIP 路由器之间是通过交换 RIP 消息（RIP Message）来实现路由信息的交换的。RIP 消息分为两种，一种是 RIP 请求（Request）消息，另一种是 RIP 响应（Response）消息。刚刚启动后的 RIP 路由器应立即向它的所有邻居发出 RIP 请求消息，以便尽快获取到关于整个 RIP 网络的路由信息；运行中的 RIP 路由器也可以随时根据需要向它的所有邻居发出 RIP 请求消息。RIP 路由器在接收到 RIP 请求消息后，应立即发出 RIP 响应消息进行回应。另外，RIP 路由器总是会每隔 30 秒钟周期性地向它的所有邻居发送 RIP 响应消息，该响应消息中携带了该路由器的 RIP 路由表中的最新路由信息。

RIP 协议有两个版本，分别为 RIP-1（RIP Version 1，RIPv1）和 RIP-2（RIP Version 2，RIPv2）。关于这两个版本的区别，我们后面会进行详细的描述。在本小节中，我们只关心 RIP-1 消息的格式。

图 8-11 显示了 RIP-1 消息的通用格式，其主要字段的解释如下。

（1）命令

该字段的长度为 8bit，取值为 1 时表示该消息是一个请求消息，取值为 2 时表示该消息是一个响应消息。

（2）版本

该字段的长度为 8bit，取值为 1 时表示该消息是一个 RIP-1 消息，取值为 2 时表示该消息是一个 RIP-2 消息。

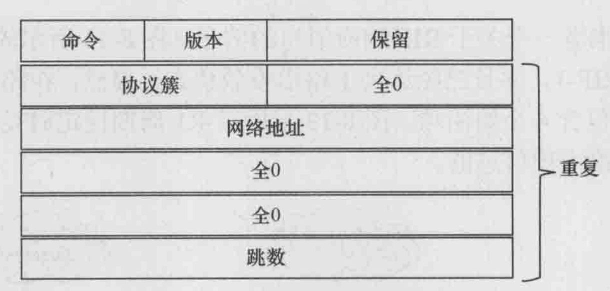


图 8-11　RIP-1 消息的通用格式

（3）协议簇

该字段的长度为 16bit，表示所使用的协议簇。对于 TCP/IP 协议簇，该字段的取值为 2。

（4）网络地址

该字段的长度为 32bit，表示一个路由项的目的网络地址。事实上，协议簇字段后面的 14 个字节都是用来表示一个路由项的目的网络地址的。由于我们使用的协议簇都是 TCP/IP，一个网络地址只需要 4 个字节来表示，所以除了这 4 个字节外，其他部分全部置 0。

（5）跳数

该字段的长度为 32bit，表示路由项的 Cost（跳数）。Cost 的最小取值是 1，最大取值是 16。如果 Cost 为 16，则表示相应的路由是一条不可达的路由。

从图 8-11 中可以看到，RIP 消息格式中的某些部分是可以多次重复的（最多可以重复 24 次），也就是说，一个 RIP 消息中最多可以包含 25 条路由信息。

RIP 请求消息可以分为两种，第一种请求消息是用来请求关于某一些指定的路由信息的，第二种请求消息是用来请求关于整个 RIP 网络的路由信息的。RIP 路由器刚刚启动之后，应立即向它的所有邻居发出第二种 RIP 请求消息。图 8-12 显示了这两种不同的 RIP-1 请求消息的差别。

请求关于若干个指定的目的网络的路由

1	1	保留	
2		0	重复
网络地址			重复
0			重复
0			重复
0			重复

请求关于所有的目的网络的路由

1	1	保留
0		0
0		
0		
0		
16		

图 8-12　两种不同的 RIP-1 请求消息

接下来，我们来看一个关于 RIP 响应消息的例子。图 8-13 所示的网络中，假设 R1、R2、R3 都运行了 RIP-1，并且已经达到了路由收敛状态。显然，在路由收敛状态下，R1 的 RIP 路由表中会包含 4 个路由项。图 8-13 显示了 R1 周期性定时发送的 RIP 响应消息是如何携带这 4 个路由项信息的。

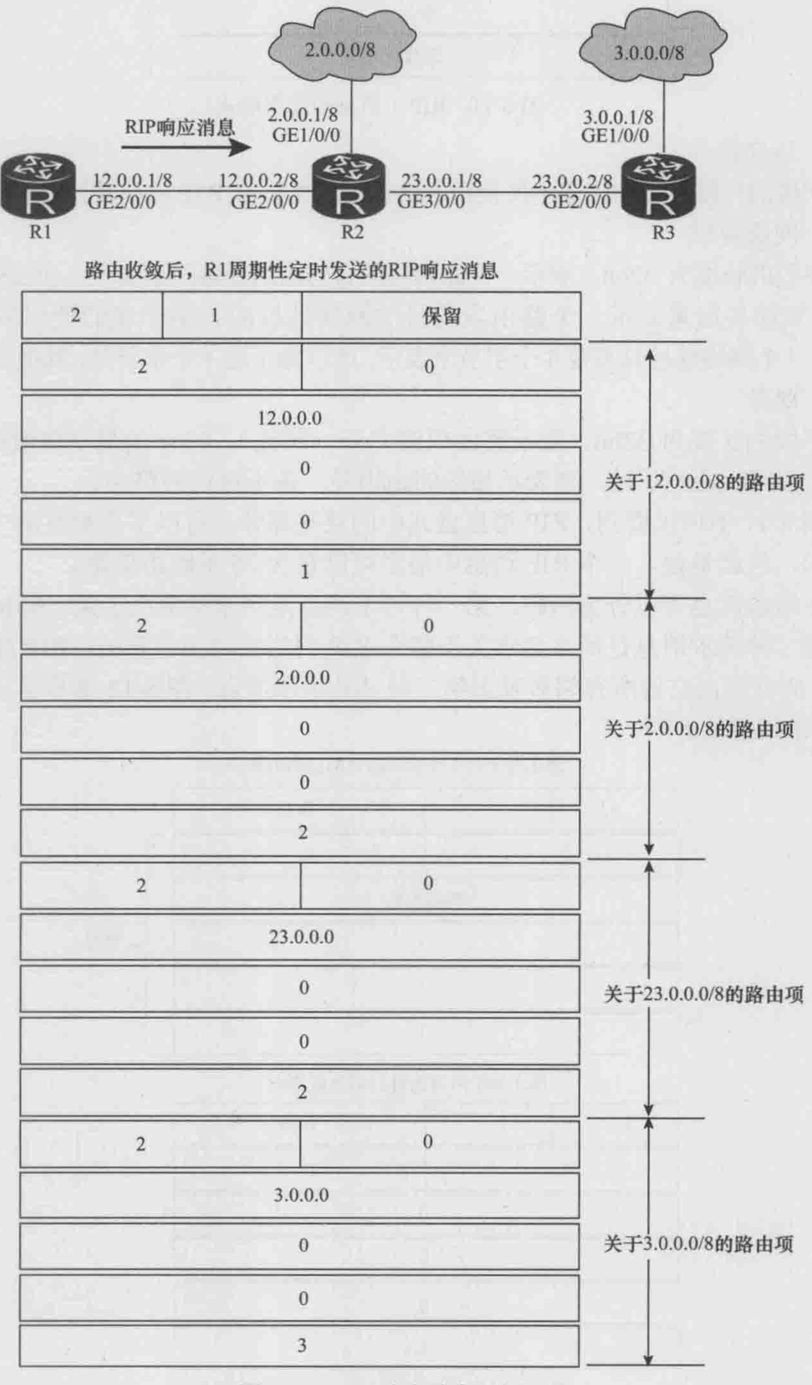


图 8-13 RIP 响应消息示例

8.2.5　RIP-1 与 RIP-2

前面曾提到，RIP 协议有两个版本，分别为 RIP-1 和 RIP-2。现在我们来看一下 RIP-2 的消息格式，如图 8-14 所示，并与图 8-11 进行比较。

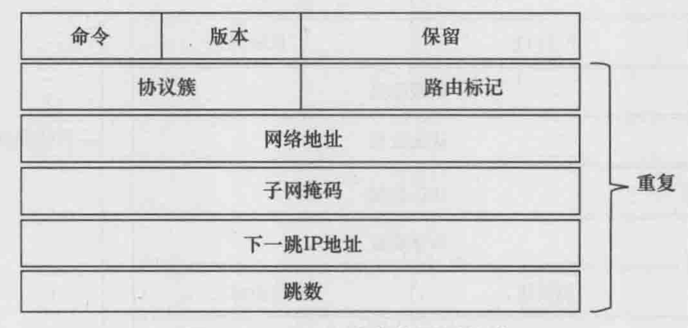


图 8-14　RIP-2 消息的通用格式

（1）命令

该字段的长度为 8bit，其含义与 RIP-1 中相同。

（2）版本

该字段的长度为 8bit，其含义与 RIP-1 中相同。取值为 2 时表示该消息是一个 RIP-2 消息。

（3）协议簇

该字段的长度为 16bit，其含义与 RIP-1 中相同。

（4）路由标记

该字段的长度为 16bit，它可以用来携带自治系统的编号等信息，从而使得 RIP-2 协议可以接收来自 BGP 协议的信息。对于该字段的具体使用场景和使用方法，我们不做深究。

（5）网络地址

该字段的长度为 32bit，其含义与 RIP-1 中相同。

（6）子网掩码

该字段的长度为 32bit，携带了网络地址字段中的 IP 地址的子网掩码。

（7）下一跳 IP 地址

该字段的长度为 32bit，表示了去往网络地址字段中的目的网络的下一跳 IP 地址。对于该字段的具体使用场景和使用方法，我们不做深究。

（8）跳数

该字段的长度为 32bit，其含义与 RIP-1 中相同。

通过对图 8-14 和图 8-11 的比较，我们不难发现，RIP-2 对 RIP-1 的功能进行了一些扩展。特别地，RIP-1 消息中不能携带子网掩码信息，而 RIP-2 消息中是可以携带子网掩码信息的，这一差别决定了 RIP-1 只能适合于有类编址的场景（采用缺省掩码），实现有类路由（Classful Routing）功能，而 RIP-2 则支持无类路由（Classless Routing），支持 VLSM、CIDR（Classless Inter-Domain Routing）等特性。

RIP-1 与 RIP-2 的另一个主要差别是，前者不能够支持认证（Authentication）功

能，而后者是可以支持认证功能的。认证功能的作用是用来对付网络中的恶意路由器所发布的一些虚假或错误的路由信息。支持认证功能时，RIP-2 的消息格式如图 8-15 所示。

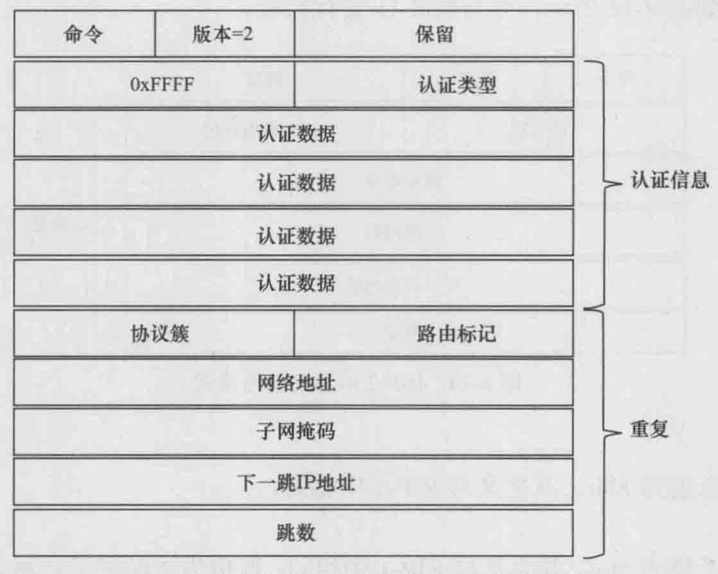


图 8-15 支持认证功能时的 RIP-2 消息格式

我们现在来看看 RIP 消息的封装情况。无论是 RIP-1 还是 RIP-2，RIP 消息都是封装在 UDP 报文中的，端口号为 520，而 UDP 报文又是封装在 IP 报文中的。运行 RIP-1 时，对于请求消息或周期性的响应消息（情况 1），IP 报文的目的地址为广播 IP 地址 255.255.255.255，源地址为发送该请求消息或响应消息的接口的 IP 地址，协议字段的值为 0x11。对于为了回应请求消息而发送的响应消息（情况 2），IP 报文的目的地址为发送请求消息的接口的 IP 地址，源地址为发送该响应消息的接口的 IP 地址，协议字段的值为 0x11。接下来，IP 报文又是封装在以太帧中的（假设路由器的接口都是以太接口）。对于情况 1，以太帧的目的地址为广播 MAC 地址 ff-ff-ff-ff-ff-ff，源地址为发送该请求消息或响应消息的接口的 MAC 地址，类型字段的值为 0x0800。对于情况 2，以太帧的目的地址为发送请求消息的接口的 MAC 地址，源地址为发送该响应消息的接口的 MAC 地址，类型字段的值为 0x0800。

运行 RIP-2 时，对于请求消息或周期性的响应消息（情况 1），IP 报文的目的地址可以为组播 IP 地址 224.0.0.9（该组播地址对应的是所有运行 RIP-2 的路由器），也可以为广播 IP 地址 255.255.255.255，源地址为发送该请求消息或响应消息的接口的 IP 地址，协议字段的值为 0x11。对于为了回应请求消息而发送的响应消息（情况 2），IP 报文的目的地址为发送请求消息的接口的 IP 地址，源地址为发送该响应消息的接口的 IP 地址，协议字段的值为 0x11。接下来，IP 报文又是封装在以太帧中的（假设路由器的接口都是以太接口）。对于情况 1，以太帧的目的地址可以为广播 MAC 地址 ff-ff-ff-ff-ff-ff，也可以为某个选定的组播 MAC 地址，源地址为发送该请求消息或响应消息的接口的 MAC 地址，类型字段的值为 0x0800。对于情况 2，以太帧的目的地址为发送请求消息

的接口的MAC地址，源地址为发送该响应消息的接口的MAC地址，类型字段的值为0x0800。

从上面的描述我们可以看到，RIP-2消息与RIP-1消息在封装过程中略有差异。由于这种差异的存在，RIP-2可以比RIP-1占用更少的设备资源。例如，在图8-16中，我们先假定R1和R2运行了RIP-1，所以R1和R2会周期性地通过它们的GE1/0/0接口发送RIP-1响应消息。由于封装了这些响应消息的帧是广播帧，所以交换机会将这些广播帧泛洪给PC。PC接收到这些广播帧后，会根据帧的类型字段的值0x0800把作为载荷数据的IP报文上送给自己网络层的IP模块。PC的IP模块发现这些IP报文的目的地址为广播IP地址255.255.255.255，于是就会根据IP报文中协议字段的值0x11将作为载荷数据的UDP报文上送给自己传输层的UDP模块。PC的UDP模块发现这些UDP报文的端口号为520，但是PC的应用层是不存在RIP模块的（因为PC不运行RIP协议），所以PC的UDP模块会丢弃这些UDP报文。

现在，在图8-16中，我们假定R1和R2运行了RIP-2，所以R1和R2会周期性地通过它们的GE1/0/0接口发送RIP-2响应消息。假设封装了这些响应消息的帧是广播帧，所以交换机会将这些广播帧泛洪给PC。PC接收到这些广播帧后，会根据帧的类型字段的值0x0800把作为载荷数据的IP报文上送给自己网络层的IP模块。假设这些IP报文的目的地址采用的是组播IP地址224.0.0.9，而PC并没有运行RIP-2，所以PC的IP模块会丢弃这些IP报文。



图8-16　RIP-2可以比RIP-1占用更少的设备处理资源

在上面的例子中我们看到，RIP协议是会占用（浪费）并不运行RIP协议的计算机的处理资源的。但是相对来讲，RIP-2要比RIP-1少占用（浪费）计算机的处理资源，这是因为计算机在网络层就可以把携带有RIP-2消息的报文丢弃，而必须在传输层才能把携带有RIP-1消息的报文丢弃。

概括地讲，相比于RIP-1，RIP-2的主要优势有以下几点。

（1）RIP-1只支持有类路由；RIP-2支持无类路由，支持VLSM、CIDR等特性。

（2）RIP-1不支持认证功能；RIP-2可以支持认证功能，因此安全性得到了提高。

（3）RIP-1不能采用组播方式发布消息；RIP-2可以采用组播方式发布消息，因此RIP-2可以比RIP-1占用更少的设备处理资源。

关于RIP-2与RIP-1的其他一些差异，这里就不赘述了。最后需要说明的是，RIP-2是可以后向兼容RIP-1的。但是，关于在实际的网络中如何混合使用RIP-1和RIP-2的

问题，我们这里不做分析和讨论。

8.2.6 RIP 定时器

RIP 协议使用了三种定时器，分别是更新定时器（Update Timer）、无效定时器（Invalid Timer）、垃圾收集定时器（Garbage Collection Timer）。

1. 更新定时器

更新定时器也称为周期定时器（Periodic Timer），每台 RIP 路由器都有一个属于自己的 RIP 更新定时器。缺省情况下，更新定时器的周期值为 30 秒。更新定时器是一个倒计时定时器，每当更新定时器的值倒计为 0 时，路由器便会向它的所有邻居发送 RIP 响应消息。注意，当路由器接收到 RIP 请求消息的时候，就会立即发送 RIP 响应消息，但这并不影响基于更新定时器的周期性 RIP 响应消息的发送。

2. 无效定时器

每台 RIP 路由器都会为自己的 RIP 路由表中的每一个路由项建立并维护一个无效定时器。无效定时器也是一个倒计时定时器。缺省情况下，无效定时器的初始值为 180 秒（更新定时器的周期值的 6 倍）。在 RIP 路由表中，一个路由项被创建时或者每次被更新时（请仔细复习 RIP 路由表的更新算法），该路由项的无效定时器的值就会被复位成初始值，然后开始倒计时。通常情况下，一个路由项每隔 30 秒钟就会被更新一次。当一个路由项的无效定时器的值倒计为 0 时，就说明该路由项已经有 180 秒的时间没有被更新了，此时路由器会认为该路由项已经变为一个无效的路由项，也就是认为该路由项所指的目的地已经变为不可达，于是路由器会将该路由的 Cost 设置为 16。

3. 垃圾收集定时器

刚才说到，当一个路由项的无效定时器的值倒计为 0 时，该路由项便成为了一个无效路由项，其 Cost 的值会被设置为 16。注意，路由器并不会立即将这个无效路由项删除掉，而是会为该无效路由项启用一个被称为垃圾收集定时器的倒数计时器。垃圾收集定时器的缺省初始值为 120 秒。在垃圾收集定时器的值倒计为 0 之前，该路由器仍然会在周期性的 RIP 响应消息中携带这条无效路由的信息，其目的是告诉它的所有邻居这条路由对于自己来说已经无效，以便邻居路由器能够及时对各自的 RIP 路由表中的相应路由项进行更新。一旦垃圾收集定时器的值倒计为 0，路由器便会将该无效路由项的所有信息（包括与该路由项对应的无效定时器和垃圾收集定时器）立即删除掉。注意，在垃圾收集定时器的值倒计为 0 之前的某一时刻，如果该无效路由被更新成为一条有效路由（即 Cost 的值被更新为小于 16），则该路由项的无效定时器的值会被复位成初始值，然后开始倒计时，而相应的垃圾收集定时器则会被删除掉。

我们现在来分析和解答一个关于 RIP 定时器的简单问题。假设在某一时刻，一台路由器的 RIP 路由表中共有 30 个路由项，其中 Cost 小于 16 的路由项有 23 个，Cost 等于 16 的路由项有 7 个，那么此刻一共有几个 RIP 定时器正在工作（计时）呢？

正确的分析和解答应该是：在此时刻，30 个无效定时器中有 23 个正在倒计时，另外 7 个已经停止计时（这 7 个无效定时器的值停留在 0），并且有 7 个垃圾收集定时器也

正在倒计时，同时还有 1 个更新定时器也在计时。所以，在此时刻，一共有 31(23+7+1=31)个 RIP 定时器正在工作。

8.2.7 RIP 网络的路由环路问题

RIP 路由表的更新算法（DV 算法）非常简单，也易于实现。但是，在某些情况下，这种算法会导致路由环路（Routing Loop）的产生。下面，我们通过一个简单的例子来说明什么是路由环路。

在图 8-17 所示的 RIP 网络中，假定路由已经收敛，那么在 R3 的 RIP 路由表中会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R3 的 GE1/0/0，下一跳 IP 地址为 3.0.0.1，Cost 为 1；R2 的 RIP 路由表中也会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R2 的 GE3/0/0，下一跳 IP 地址为 23.0.0.2，Cost 为 2。

现在，假设 R3 去往 3.0.0.0/8 的物理链路因为某种原因突然中断，导致 R3 的 GE1/0/0 接口无法正常工作。R3 在检测到这一故障后，会立即将自己 RIP 路由表中去往目的网络 3.0.0.0/8 的路由项的 Cost 设置为 16，表示 3.0.0.0/8 对于 R3 来说已经变为不可达了（也就是该路由已经变成了无效路由）。然而，就在 R3 等待着准备将这条无效路由的信息随下一个周期性的响应消息发送给 R2 时，却收到了 R2 发送过来的关于 3.0.0.0/8 的路由信息。根据 DV 算法，R3 会将自己的 RIP 路由表中那条去往 3.0.0.0/8 的无效路由重新更新成为有效路由，出接口更新为 R3 的 GE2/0/0，下一跳 IP 地址更新为 23.0.0.1，Cost 更新为 3。也就是说，R3 会认为自己虽然无法“直达” 3.0.0.0/8，但是可以通过 R2 间接地到达 3.0.0.0/8。此时我们会发现，R3 和 R2 的路由表中都各自存在一条去往 3.0.0.0/8 的有效路由，且下一跳都是指向对方的，这样便产生了路由环路。如果 R2 或 R3 需要转发一个去往 3.0.0.0/8 的 IP 报文，那么该 IP 报文将在 R2 和 R3 之间被转来转去。

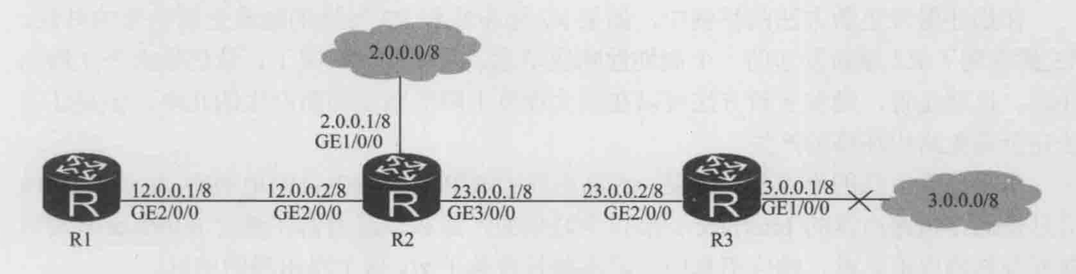


图 8-17 RIP 路由环路问题

显然，路由环路是有损于网络的正常工作的。为此，RIP 协议提供了 3 种不同的方法来解决这一问题。这 3 种方法分别是：触发更新（Triggered Update）、水平分割（Split Horizons）、毒性逆转（Poison Reverse）。

1. 触发更新

所谓触发更新，就是指当 RIP 路由表中的某些路由项的内容发生改变时，路由器应立即向它的所有邻居发布响应消息，而不要等待更新定时器所规定的下一个响应消息的发送时刻。另外，为了减少带宽及路由器处理资源的消耗，触发更新的响应消息中只需包含路由信息发生了改变的路由项。

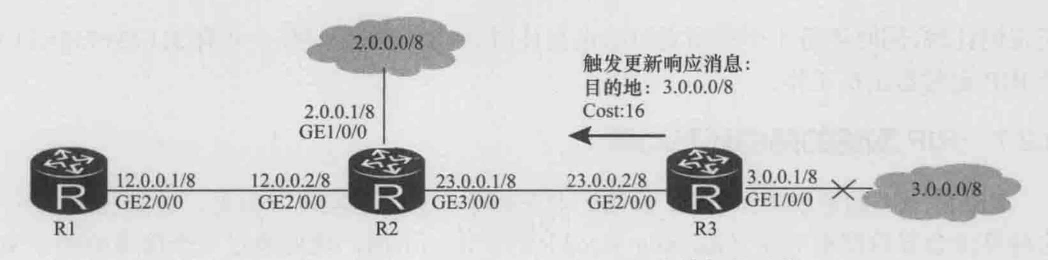


图 8-18 触发更新

图 8-18 所示的网络与图 8-17 所示的网络为同一网络。在图 8-18 中，R3 通过配置而具备了触发更新功能。路由收敛时，R3 的 RIP 路由表中会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R3 的 GE1/0/0，下一跳 IP 地址为 3.0.0.1，Cost 为 1；R2 的 RIP 路由表中也会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R2 的 GE3/0/0，下一跳 IP 地址为 23.0.0.2，Cost 为 2。

现在，假设 R3 去往 3.0.0.0/8 的物理链路因为某种原因突然中断，导致 R3 的 GE1/0/0 接口无法正常工作。R3 在检测到这一故障后，会立即将自己 RIP 路由表中去往目的网络 3.0.0.0/8 的路由项的 Cost 设置为 16。因为 R3 的 RIP 路由表中关于 3.0.0.0/8 的路由项的内容发生了改变，所以 R3 会立即向 R2 发送关于 3.0.0.0/8 的路由更新消息（触发更新消息）。R2 收到 R3 发来的关于 3.0.0.0/8 的路由更新消息后，会根据 DV 算法立即将自己的 RIP 路由表中 3.0.0.0/8 这个路由项的 Cost 更新为 16。尽管 R2 和 R3 接下来还会周期性地相互发送 RIP 响应消息，但是根据 DV 算法，它们的 RIP 路由表中 3.0.0.0/8 这个路由项的 Cost 都会保持为 16，即 R2 和 R3 都会认为 3.0.0.0/8 是不可达的。如果 R2 或 R3 需要转发一个去往 3.0.0.0/8 的 IP 报文，那么该 IP 报文不会在 R2 和 R3 之间被转来转去，而是会被 R2 或 R3 直接丢弃掉。

2. 水平分割

在描述触发更新方法的举例中，如果 R2 还未收到 R3 发送的触发更新消息的时候，R3 就收到了 R2 最新发送的一个周期性响应消息，在这样的情况下，就仍然会产生路由环路。也就是说，触发更新方法可以在很大程度上降低路由环路产生的几率，但是还无法完全避免路由环路的产生。

水平分割方法的原理是，如果一台路由器的 RIP 路由表中目的地/掩码为 z/y 的路由信息是通过该路由器的 Interface-x 接口学习来的，那么该路由器在通过 Interface-x 接口向外发送响应消息时，响应消息中一定不要包含关于 z/y 这个路由项的信息。

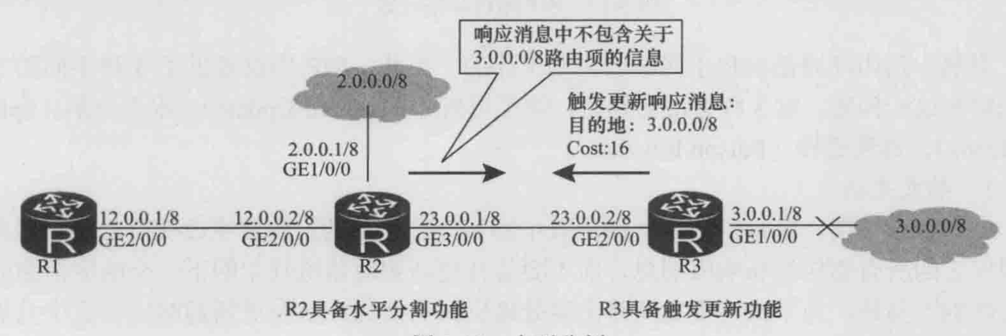


图 8-19 水平分割

图 8-19 所示的网络与图 8-18 所示的网络为同一网络。在图 8-19 中，R3 通过配置而具备了触发更新功能，R2 通过配置而具备了水平分割功能。路由收敛时，R3 的 RIP 路由表中会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R3 的 GE1/0/0，下一跳 IP 地址为 3.0.0.1，Cost 为 1；R2 的 RIP 路由表中也会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R2 的 GE3/0/0，下一跳 IP 地址为 23.0.0.2，Cost 为 2。注意，由于 R2 的 RIP 路由表中 3.0.0.0/8 这个路由项的出接口是 R2 的 GE3/0/0，这就说明该路由项是通过 R2 的 GE3/0/0 接口学习来的。因此，R2 在通过自己的 GE3/0/0 接口向外发送响应消息时，响应消息中一定不会包含关于 3.0.0.0/8 这条路由的信息。

现在，假设 R3 去往 3.0.0.0/8 的物理链路因为某种原因突然中断，导致 R3 的 GE1/0/0 接口无法正常工作。R3 在检测到这一故障后，会立即将自己 RIP 路由表中去往目的网络 3.0.0.0/8 的路由项的 Cost 设置为 16，并且会立即向 R2 发送关于 3.0.0.0/8 的路由更新消息（触发更新消息）。分析表明，即使在 R2 收到 R3 发送的触发更新消息之前，R3 就收到了 R2 最新发送的一个周期性响应消息（注意，这个响应消息中是不包含关于 3.0.0.0/8 的路由信息的），也不会导致路由环路的产生。R2 在接收到 R3 发来的关于 3.0.0.0/8 的触发更新消息后，会根据 DV 算法立即将自己的 RIP 路由表中 3.0.0.0/8 这个路由项的 Cost 更新为 16。尽管 R2 和 R3 接下来还会周期性地相互发送 RIP 响应消息，但是根据 DV 算法，它们的 RIP 路由表中 3.0.0.0/8 这个路由项的 Cost 都会保持为 16。当然，垃圾收集定时器超时的时候，该路由项会被从 RIP 路由表中彻底删除。

3. 毒性逆转

毒性逆转方法的原理是，如果一台路由器的 RIP 路由表中目的地/掩码为 z/y 的路由信息是通过该路由器的 Interface-x 接口学习来的，那么该路由器在通过 Interface-x 接口向外发送响应消息时，响应消息中仍然需要包含 z/y 这个路由项，但这个路由项的 Cost 总是设置为 16。

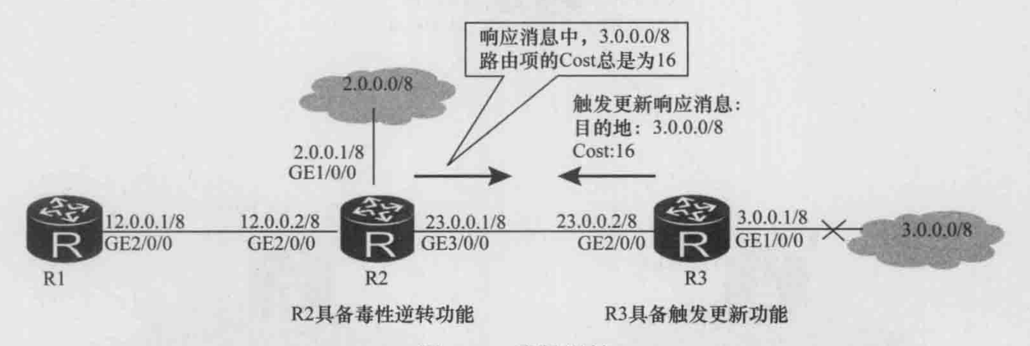


图 8-20　毒性逆转

图 8-20 所示的网络与图 8-18 所示的网络为同一网络。在图 8-20 中，R3 通过配置而具备了触发更新功能，R2 通过配置而具备了毒性逆转功能。路由收敛时，R3 的 RIP 路由表中会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R3 的 GE1/0/0，下一跳 IP 地址为 3.0.0.1，Cost 为 1；R2 的 RIP 路由表中也会存在一条目的网络为 3.0.0.0/8 的路由，出接口为 R2 的 GE3/0/0，下一跳 IP 地址为 23.0.0.2，Cost 为 2。注意，由于 R2 的 RIP 路由表中 3.0.0.0/8 这个路由项的出接口是 R2 的 GE3/0/0，这就说明该路由项是通过 R2

的 GE3/0/0 接口学习来的。因此，R2 在通过自己的 GE3/0/0 接口向外发送响应消息时，响应消息中仍然需要包含 3.0.0.0/8 这个路由项，但这个路由项的 Cost 总是设置为 16。

现在，假设 R3 去往 3.0.0.0/8 的物理链路因为某种原因突然中断，导致 R3 的 GE1/0/0 接口无法正常工作。R3 在检测到这一故障后，会立即将自己 RIP 路由表中去往目的网络 3.0.0.0/8 的路由项的 Cost 设置为 16，并且会立即向 R2 发送关于 3.0.0.0/8 的路由更新消息（触发更新消息）。分析表明，即使在 R2 收到 R3 发送的触发更新消息之前，R3 就收到了 R2 最新发送的一个周期性响应消息（注意，这个响应消息中 3.0.0.0/8 这个路由项的 Cost 为 16），也不会导致路由环路的产生。

毒性逆转方法和水平分割方法都能避免路由环路的产生，二者的工作原理也非常相似。但需要注意的是，这两种方法是互斥的。也就是说，RIP 路由器可以具备水平分割功能，也可以具备毒性逆转功能，但是不可能同时具备这两种功能（请读者朋友们解释一下为什么会是这样）。在实际应用中，通常会在 RIP 路由器上配置触发更新功能（触发更新功能除了能够降低路由环路产生的几率外，还能够加快路由收敛速度），然后在水平分割和毒性逆转中选择配置其中的一种功能。

8.2.8 RIP 基本配置示例

如图 8-21 所示，某公司有 3 台路由器，其中 R2 为公司总部的路由器，R1 和 R3 分别为分支机构 A 和分支机构 B 的路由器。所有的路由器都需要运行 RIP 协议，以实现整个网络的互通性。

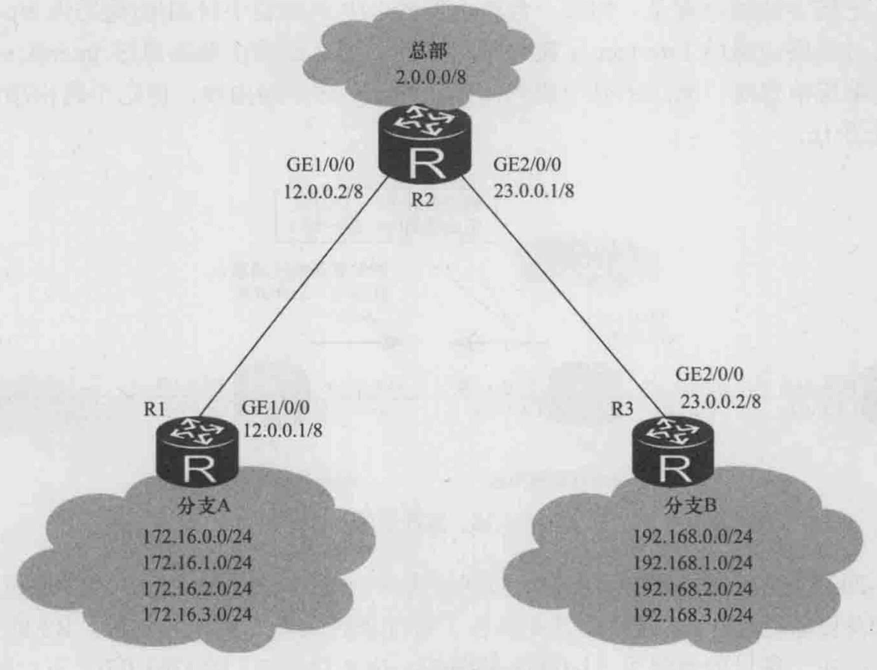


图 8-21 RIP 基本配置示例

1. 配置思路

在各路由器上启动 RIP 进程，在 RIP 进程中发布网段信息。

2. 配置步骤

要在路由器上配置 RIP，必须首先进入系统视图，然后执行命令 **rip** [*process-id*]，以启动 RIP 进程，并进入 RIP 视图。执行该命令时，如果不输入 *process-id*（该参数表示 RIP 进程编号）的值，则 *process-id* 默认取值为 1。

#配置 R1。

```
<R1> system-view
[R1] rip
[R1-rip-1]
```

#配置 R2。

```
<R2> system-view
[R2] rip
[R2-rip-1]
```

#配置 R3。

```
<R3> system-view
[R3] rip
[R3-rip-1]
```

启动 RIP 进程之后，还需要通过 **network** *network-address* 命令发布指定的网段，其中 *network-address* 必须是一个自然网段的网络地址。

#配置 R1。

```
[R1-rip-1] network 12.0.0.0
[R1-rip-1] network 172.16.0.0
```

#配置 R2。

```
[R2-rip-1] network 12.0.0.0
[R2-rip-1] network 23.0.0.0
[R2-rip-1] network 2.0.0.0
```

#配置 R3。

```
[R3-rip-1] network 23.0.0.0
[R3-rip-1] network 192.168.0.0
[R3-rip-1] network 192.168.1.0
[R3-rip-1] network 192.168.2.0
[R3-rip-1] network 192.168.3.0
```

为了对所做的配置进行确认，我们可以使用 **display rip** [*process-id*]命令查看 RIP 的当前运行状态及配置信息。例如，我们可以在 R1 上执行该命令。

```
<R1> display rip
  Public VPN-instance
    RIP process : 1
        RIP version : 1
        Preference : 100
......
        Update time     : 30 sec   Age time    : 180 sec
        Garbage-collect time : 120 sec
......
        Networks :
        12.0.0.0          172.16.0.0
```

……

从上面的回显信息中，我们可以看到如下信息。

"RIP process：1"表示 RIP 的进程编号为 1。

"RIP version：1"表示运行的是 RIPv1。

“Preference：100”表示 RIP 的协议优先级的值为 100。

“Update time：30 sec”表示更新定时器的周期值为 30 秒。

“Age time：180 sec”表示无效定时器的初始值为 180 秒。无效定时器也称为老化定时器（Aging Timer）。

“Garbage-collect time：120 sec”表示垃圾收集定时器的初始值为 120 秒。

为了确认 R1 是否可以收到 R2 发布的路由，我们可以在 R1 上使用 **display rip** *process-id* **route** 命令来查看 R1 从其他路由器那里学习到的所有 RIP 路由信息。

```
<R1> display rip 1 route
 Route Flags:   R - RIP
                A – Aging, G - Garbage-collect
 ----------------------------------------------------------------------
Destination/Mask      Nexthop       Cost       Tag       Flags       Sec
 2.0.0.0/8            12.0.0.2      1          0         RA          15
 23.0.0.0/8           12.0.0.2      1          0         RA          15
 192.168.0.0/24       12.0.0.2      2          0         RA          15
 192.168.1.0/24       12.0.0.2      2          0         RA          15
 192.168.2.0/24       12.0.0.2      2          0         RA          15
 192.168.3.0/24       12.0.0.2      2          0         RA          15
```

从上面的回显信息中，我们看到 R1 已经学习到了关于 2.0.0.0/8，23.0.0.0/8，192.168.0.0/24，192.168.1.0/24，192.168.2.0/24，192.168.3.0/24 这些非直连路由的信息。

8.2.9 练习题

1.（多选）下列描述中正确的是？（　）

A．RIP 协议是一种静态路由协议

B．RIP 协议是一种 EGP（Exterior Gateway Protocol）

C．OSPF 协议是一种 IGP（Interior Gateway Protocol）

D．RIP 协议是一种基于 DV（Distance Vector）算法的路由协议

2.（单选）一个 RIP 响应消息中最多可以包含多少个路由项的信息？（　）

A．1 个　　B．15 个　　C．25 个　　D．35 个

3.（多选）以下选项中哪些不是解决 RIP 路由环路的方法？（　）

A．触发更新　　B．水平分割　　C．毒性反转　　D．DV 算法

4.（单选）一个 RIP 响应消息在封装进 UDP 报文之前，其长度不可能超过多少字节？（　）

A．204 字节　　B．304 字节　　C．404 字节　　D．504 字节

5.（多选）与 RIP-1 相比，以下哪些选项是 RIP-2 的优势？（　）

A．RIP-2 可以使用组播方式发送 RIP 消息

B．RIP-2 支持认证功能

C．IP-2 不仅可以将跳数作为路由开销的定义，还可以将带宽作为路由开销的定义

D．RIP-2 支持无类路由

6.（多选）下列描述中正确的是？（　）

A．一台 RIP 路由器可以同时具备触发更新功能和水平分割功能

B．一台 RIP 路由器可以同时具备触发更新功能和毒性逆转功能

C. 一台 RIP 路由器可以同时具备毒性逆转功能和水平分割功能

D. 一台 RIP 路由器可以同时具备触发更新功能、水平分割功能、毒性逆转功能

8.3 OSPF 协议

与 RIP 协议一样，OSPF（Open Shortest Path First）协议也是一种 IGP。通常，我们把一个以 OSPF 协议作为其 IGP 的自治系统称为一个 OSPF 网络。

然而，OSPF 协议的复杂程度远远大于 RIP 协议。曾经，笔者只花了半天的时间便学习完了 RIP 协议的内容，但却花了整整两周的时间才学习完 OSPF 协议的内容。鉴于本书所规划的知识范围和程度所限，我们接下来只粗略地介绍一下 OSPF 的基本原理和和常见术语，对于其具体的原理细节和工作过程不做深究。

8.3.1 OSPF 的基本原理

在讲述 RIP 协议的基本原理时，我们曾提到了一个游戏活动，内容如下。

"假设在一个教室里坐满了新同学，坐中间的每个同学有前、后、左、右 4 个邻居，坐边上的同学有 3 个邻居，坐角落的同学有两个邻居。游戏开始前，假定每个同学都只知道自己的所有邻居的姓名，也就是说，每个同学的'记忆库'中只有自己的几个邻居的姓名。游戏开始后，每个同学都周期性地把自己最新的记忆库中的所有姓名悄悄地告诉给自己的所有邻居（每个同学只能听见邻居对自己说的话），同时不停地把自己从邻居那里听来的姓名装进自己的记忆库。游戏持续了足够长的时间后，我们会发现每个同学的记忆库中的内容都不再发生变化，并且都包含了全班所有同学的姓名。这个游戏的过程虽然非常简单，但它正好体现了 RIP 协议的基本原理。"

现在，我们对游戏活动的规则做一下改变。游戏开始前，仍然假定每个同学都只知道自己的所有邻居的姓名。游戏开始后，每个同学都尽快一次性地大声地对全班同学说出自己所有邻居的姓名（假设教室里的声音无论有多么嘈杂，同学们都能听见、能分辨、能记住这些声音的内容）。显然，当最后一个同学说完之后，每个同学的记忆库中便有了全班所有同学的姓名。这个游戏的过程也非常简单，但它正好体现了 OSPF 协议的基本原理。

通过比较，我们不难发现前后两个游戏（分别称为游戏 1 和游戏 2）的主要差别在于以下 3 个方面。

（1）游戏 1 中，每个同学只对邻居说悄悄话，每个同学也只能听见邻居对自己说的话；游戏 2 中，每个同学都是大声说话，说话内容全班每个同学都能听见。

（2）游戏 1 中，每个同学说话的内容是自己记忆库中最新拥有的所有同学的姓名；游戏 2 中，每个同学说话的内容只是自己所有邻居的姓名。

（3）游戏 1 中，每个同学都会周期性地、反反复复啰啰嗦嗦地听来讲去；游戏 2 中，整个说话的过程一次性便可完成。

如果将上述 3 点转换成网络技术的语言，便可得到这样一句话：在 RIP 协议中，路由器会将自己所知道的关于整个网络的路由信息周期性地发送给所有的邻居路由器；在 OSPF 协议中，路由器会将自己的链路状态信息一次性地泛洪（Flooding）给所有其他的路由器。

需要说明的是，OSPF协议中引入了Area（区域）的概念。游戏1中，整个教室相当于整个RIP网络；但在游戏2中，整个教室只相当于OSPF网络的一个Area。关于Area的概念，我们后面会进行描述和解释。

另外，请读者不要纠结于游戏1和游戏2这两个比喻的细节问题。使用这两个比喻的目的，只是希望读者能够对RIP和OSPF有一个更加直观、更加形象、更加感性的认识而已。对于比喻中的不恰当之处，敬请读者忽略之。

8.3.2 OSPF与RIP的比较

OSPF是一种基于链路状态（Link-State）的路由协议，而RIP则是一种基于距离矢量的路由协议，这是二者之间最根本性的差别。关于什么是链路状态，我们后面会进行描述和解释。

RIP协议中，路由器之间是以一种“传话”的方式来传递有关路由的信息。OSPF协议中，路由器之间可以以一种“宣告”的方式来传递有关路由的信息。OSPF网络的路由收敛时间明显小于RIP网络的路由收敛时间。

RIP是一种“嘈杂”的路由协议。路由收敛之后，RIP网络中仍然会持续性地存在大量的RIP协议报文的流量。OSPF是一种“安静”的路由协议。路由收敛之后，OSPF网络中OSPF协议报文的流量很少。协议报文的流量越小，对网络带宽资源的占用就越少。

RIP协议是以UDP作为其传输层协议的，RIP报文是封装在UDP报文中的。OSPF没有传输层协议，OSPF报文是直接封装在IP报文中的。我们知道，UDP通信或IP通信都是一种无连接、不可靠的通信方式。RIP也好，OSPF也罢，其协议报文传输的可靠性机制都是由协议本身提供的。

RIP报文只有两种，一种是RIP请求报文，另一种是RIP响应报文。OSPF报文有5种，分别是Hello报文（Hello Packet）、数据库描述报文（Database Description Packet，DD Packet）、链路状态请求报文（Link-State Request Packet，LSR Packet）、链路状态更新报文（Link-State Update Packet，LSU Packet）、链路状态确认报文（Link-State Acknowledgement Packet，LSAck Packet）。

RIP协议只能以“跳数”来作为路由开销的定义。在OSPF协议中，理论上可以采用任何参量，或者若干参量的组合来作为路由开销的定义。例如，OSPF可以采用链路的带宽来定义路由开销，也可以采用链路的延迟时间来定义路由开销，还可以采用链路的“成本”来定义路由开销，如此等等。但在实际中，最常见的是采用链路的带宽来定义路由开销。

RIP和OSPF都是IETF制定的开放性标准协议。OSPF也有两个版本，OSPFv1和OSPFv2。但是，OSPFv1在其正式发布之前的试验阶段就夭折了，所以目前实际网络中所使用的都是OSPFv2。与RIPv2一样，OSPF也是一种无类路由协议，支持VLSM、CIDR等特性。与RIPv2一样，OSPF也支持认证功能。

OSPF网络具有区域化的结构，RIP网络没有这种结构。OSPF网络中，路由器有角色之分，不同角色的路由器具有不同功能和作用。RIP网络中的路由器是没有角色之分的。OSPF网络中，每台路由器都有一个独一无二的路由器身份号（Router Identification，Router-ID）。RIP网络中，路由器是没有Router-ID的。

RIP协议和OSPF协议在实际中的应用都非常广泛。但是需要注意的是，RIP协议

只适合用在小型网络中，而 OSPF 则适用于任何规模的网络。

一言以蔽之，OSPF 协议在各个方面都是优于 RIP 协议的，如果不考虑协议复杂程度的话。

8.3.3　OSPF 的区域化结构

一个 OSPF 网络可以被划分成多个区域（Area）。如果一个 OSPF 网络只包含一个区域，则这样的 OSPF 网络称为单区域 OSPF 网络；如果一个 OSPF 网络包含了多个区域，则这样的 OSPF 网络称为多区域 OSPF 网络。

在 OSPF 网络中，每一个区域都有一个编号，称为 Area-ID。Area-ID 是一个 32bit 的二进制数，但通常也可以用十进制数来表示。Area-ID 为 0 的区域称为骨干区域（Backbone Area），否则称为非骨干区域。单区域 OSPF 网络只包含一个区域，这个区域必须是骨干区域。多区域 OSPF 网络中，除了有一个骨干区域外，还有若干个非骨干区域，并且每一个非骨干区域都需要与骨干区域直接相连（采用 Virtual Link 技术时，非骨干区域虽然没有与骨干区域直接相连，但在逻辑上仍然是与骨干区域直接相连的），但非骨干区域之间是不允许直接相连的。也就是说，非骨干区域之间的通信必须要通过骨干区域中转才能进行。

图 8-22 所示的 OSPF 网络总共包含了 4 个区域，其中 Area 0 才是骨干区域。需要注意的是，R9 的上面那个接口是属于 Area 2 的，R9 的下面那个接口是属于 Area 0 的。类似地，R10 的上面两个接口是属于 Area 3 的，R10 的下面两个接口是属于 Area 0 的；R1 的上面那个接口是属于 Area 0 的，R1 的下面 3 个接口是属于 Area 1 的。

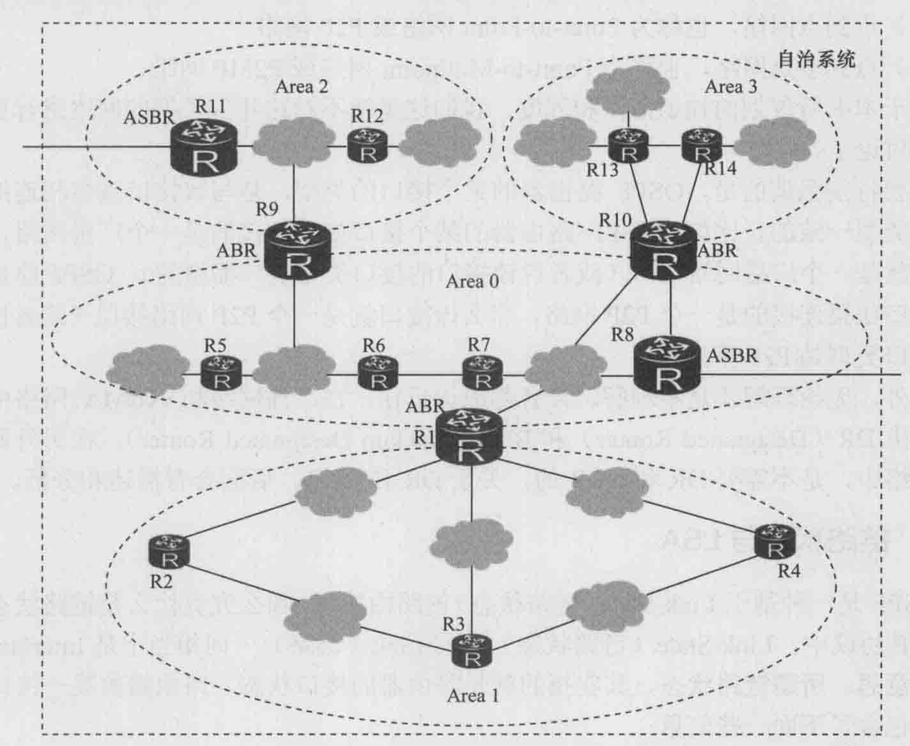


图 8-22　OSPF 的区域化结构

OSPF 网络中，如果一台路由器的所有接口都属于同一个区域，则这样的路由器被称为内部路由器（Internal Router）。图 8-22 所示的 OSPF 网络中，Area 0 的内部路由器

有 R5、R6、R7、R8。Area 1 的内部路由器有 R2、R3、R4。Area 2 的内部路由器有 R11 和 R12，Area 3 的内部路由器有 R13 和 R14。

OSPF 网络中，如果一台路由器包含有属于 Area 0 的接口，则这样的路由器被称为骨干路由器（Backbone Router）。图 8-22 所示的 OSPF 网络中，总共有 7 个骨干路由器，分别是 R5、R6、R7、R8、R1、R9、R10。

OSPF 网络中，如果一台路由器的某些接口属于 Area 0，其他接口属于别的区域，则这样的路由器被称为 ABR（Area Border Router，区域边界路由器）。图 8-22 所示的 OSPF 网络中，总共有 3 个 ABR，分别是 R1、R9、R10。

OSPF 网络中，如果一台路由器是与本 OSPF 网络（本自治系统）之外的网络相连的，并且可以将外部网络的路由信息引入进本 OSPF 网络（本自治系统），则这样的路由器被称为 ASBR（Autonomous System Boundary Router，自治系统边界路由器）。图 8-22 所示的 OSPF 网络中，总共有两个 ASBR，一个是 R8，另一个是 R11。

8.3.4 OSPF 支持的网络类型

OSPF 支持的网络类型，是指 OSPF 能够支持的二层网络的类型。OSPF 能够支持的网络类型有以下几种。

（1）广播网络，也称为 Broadcast 网络。例如，以太网便是一种典型的 Broadcast 网络。

（2）NBMA（Non-Broadcast Multi-Access）网络。

（3）点到点网络，也称为 Point-to-Point 网络或 P2P 网络。

（4）点到多点网络，也称为 Point-to-Mutipoint 网络或 P2MP 网络。

鉴于本书所规划的知识范围和程度，我们这里就不对这几种类型的网络进行更多的分析和讨论了。

需要特别强调的是，OSPF 路由器的某个接口的类型，是与该接口直接相连的二层网络的类型一致的。比如，OSPF 路由器的某个接口如果连接的是一个广播网络，那么该接口就是一个广播网络接口（或者说该接口的接口类型是广播型的）；OSPF 路由器的某个接口如果连接的是一个 P2P 网络，那么该接口就是一个 P2P 网络接口（或者说该接口的接口类型是 P2P 型的）。

另外，无论理解还是不理解，读者都应该记住，在广播网络和 NBMA 网络中，需要选举出 DR（Designated Router）和 BDR（Backup Designated Router）。在另外两种类型的网络中，是不需要 DR 和 BDR 的。关于 DR 和 BDR，后面会有描述和分析。

8.3.5 链路状态与 LSA

OSPF 是一种基于 Link-State（链路状态）的路由协议，那么究竟什么是链路状态呢？在 OSPF 协议中，Link-State（链路状态）中的 Link（链路）一词相当于是 Interface（接口）的意思。所谓链路状态，其实指的就是路由器的接口状态。路由器的某一接口的状态主要包含了下面一些信息。

（1）该接口的 IP 地址及掩码。

（2）该接口所属区域的 Area-ID。

（3）该接口所属的路由器的 Router-ID。

（4）该接口的接口类型（也就是该接口所连的二层网络的类型，如广播型、NBMA 型、点到点型、点到多点型）。

（5）该接口的接口开销（通常会以接口带宽来定义接口开销；带宽越大，开销越小）。

（6）该接口所属的路由器的 Router Priority（这个参数是用来选举 DR 和 BDR 的）。

（7）该接口所连的二层网络中的 DR。

（8）该接口所连的二层网络中的 BDR。

（9）该接口的 HelloInterval（该接口发送 Hello 报文的间隔时间）。

（10）该接口的 RouterDeadInterval（该时间参数称为路由器失效时间。如果该接口在这个时间范围内没有接收到某个邻居路由器发来的 Hello 报文，则认为那个邻居路由器已经失效）。

（11）该接口的所有邻居路由器。

（12）该接口的认证类型。

（13）该接口的密钥。

（14）……

OSPF 是一种基于链路状态的路由协议，其核心思想就是，每台路由器都将自己的各个接口的接口状态（即链路状态）共享给其他路由器。在此基础上，每台路由器就可以根据自己的各个接口的接口状态已及其他路由器各个接口的接口状态计算出从自己去往各个目的地的路由。

那么，LSA 又是什么意思呢？LSA 是 Link-State Advertisement 的缩写词，直译为“链路状态通告”。LSA 有十几种类型（Type），内容如下。

（1）Type-1 LSA（也称为 Router LSA）。

（2）Type-2 LSA（也称为 Network LSA）。

（3）Type-3 LSA（也称为 Network Summary LSA）。

（4）Type-4 LSA（也称为 ASBR Summary LSA）。

（5）Type-5 LSA（也称为 AS External LSA）。

（6）……

简单地说，LSA 是链路状态信息的主要载体，链路状态信息主要是包含在 LSA 中并通过 LSA 的通告（泛洪）来实现共享的。需要说明的是，不同类型的 LSA 中所包含的链路状态的内容是不同的；不同类型的 LSA 的功能和作用也是不同的；不同类型的 LSA 的通告（泛洪）范围也是不同的；不同角色的路由器能够产生的 LSA 的类型也是不同的，如下所示。

（1）Type-1 LSA：每台路由器都会产生。Type-1 LSA 用来描述路由器各个接口的接口类型、IP 地址、开销值等信息。一个 Type-1 LSA 只能在产生它的 Area 内泛洪，不能泛洪到其他 Area。

（2）Type-2 LSA：它是由 DR 产生的，主要用来描述该 DR 所在的二层网络的网络掩码以及该二层网络中总共包含了哪些路由器。一个 Type-2 LSA 只能在产生它的 Area 内泛洪，不能泛洪到其他 Area。

（3）Type-3 LSA：它是由 ABR 产生的。ABR 路由器将自己所在的多个 Area 中的 Type-1 和 Type-2 LSA 转换为 Type-3 LSA，这些 Type-3 LSA 描述了 Area 之间的路由信息。Type-3 LSA 可以泛洪到整个自治系统（整个 OSPF 网络）内部，但是不能泛洪到 Totally

Stub Area 和 Totally Not-So-Stubby Area（Totally Stub Area 和 Totally Not-So-Stubby Area 的概念已经超出了本书的知识范围）。

（4）Type-4 LSA：它是由 ASBR 所在 Area 的 ABR 产生的，用来描述去往 ASBR 的路由信息。Type-4 LSA 可以泛洪到整个自治系统（整个 OSPF 网络）内部，但是不能泛洪到 Stub Area（Stub Area 的概念已经超出了本书的知识范围）、Totally Stub Area、Not-So-Stubby Area（Not-So-Stubby Area 的概念已经超出了本书的知识范围）和 Totally Not-So-Stubby Area。

（5）Type-5 LSA：它是由 ASBR 产生的，用来描述去往自治系统外部的路由。Type-5 LSA 可以泛洪到整个自治系统（整个 OSPF 网络）内部，但是不能泛洪到 Stub Area、Totally Stub Area、Not-So-Stubby Area 和 Totally Not-So-Stubby Area。

（6）……

8.3.6 OSPF 报文的类型

如图 8-23 所示，OSPF 的协议报文（简称 OSPF 报文）是直接封装在 IP 报文中的，IP 报文头部中的协议字段的值必须为 89。

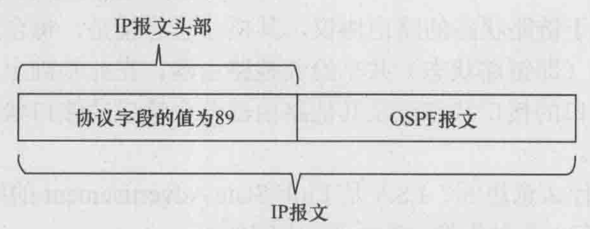


图 8-23　OSPF 报文是直接封装在 IP 报文中的

OSPF 报文本身有 5 种类型，分别是 Hello 报文、DD 报文、LSR 报文、LSU 报文、LSAck 报文，如图 8-24 所示。从图 8-24 中我们还可以看到，各种不同类型的 LSA 其实只是包含在 LSU 报文中的。其他类型的 OSPF 报文中虽然没有携带 LSA，但是仍然会携带一些链路状态信息，当然也会携带一些其他的协议信息。

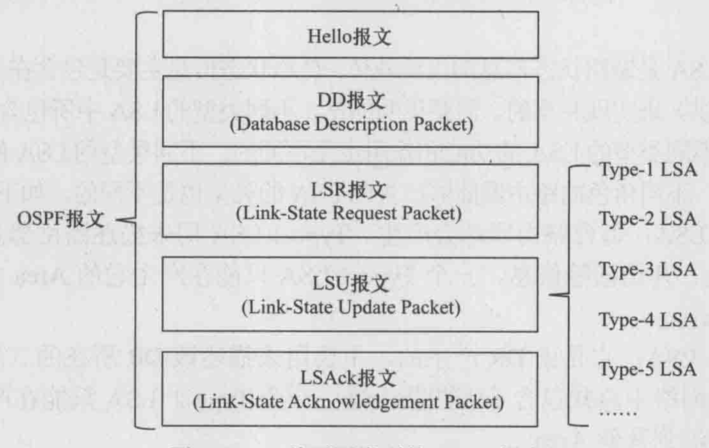


图 8-24　5 种不同类型的 OSPF 报文

鉴于本书的知识范围所限，我们将省去对这 5 种类型的 OSPF 报文的具体分析和讨论，而只是简单地了解一下 Hello 报文中所携带的信息。

路由器的某一接口所发送的 Hello 报文中主要携带了下面一些信息。

（1）OSPF 的版本号。

（2）该接口所属的路由器的 Router-ID。

（3）该接口所属区域的 Area-ID。

（4）该接口的认证类型。

（5）该接口的密钥。

（6）该接口的 IP 地址的子网掩码。

（7）该接口的 HelloInterval（该接口发送 Hello 报文的间隔时间）。

（8）该接口所属的路由器的 Router Priority（这个参数是用来选举 DR 和 BDR 的）。

（9）该接口的 RouterDeadInterval。

（10）该接口所连的二层网络中的 DR。

（11）该接口所连的二层网络中的 BDR。

（12）该接口的所有邻居路由器。

（13）……

8.3.7　单区域 OSPF 网络

图 8-25 显示的是一个单区域 OSPF 网络，整个网络只有 Area 0，该 Area 0 也就是整个自治系统。在这个 OSPF 网络中，没有 ABR，假设也没有 ASBR。我们将以这个网络为例子来简要地描述一下单区域 OSPF 网络的工作过程。

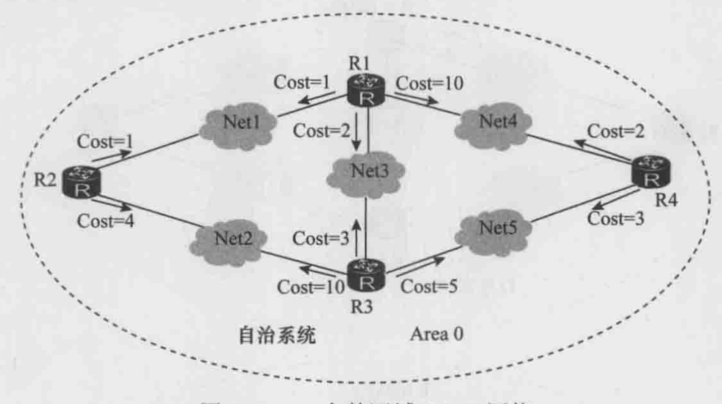


图 8-25　一个单区域 OSPF 网络

1. 链路状态数据库

图 8-25 中，每台路由器都会产生 Type-1 LSA，并向整个 Area 0 泛洪。另外，具有 DR 角色的路由器还会产生 Type-2 LSA，并向整个 Area 0 泛洪。根据我们前面所学的知识可知，整个 Area 0 中就只存在这两种类型的 LSA，不再有其他类型的 LSA 存在。

每台路由器将所有接收到的 LSA 以及自己产生的 LSA 集中在一起，便得到了一个数据库，我们把这样的数据库称为 LSDB（Link-State Dabebase，链路状态数据库）。显然，一个 LSDB 其实就是若干条 LSA 的集合。因为 LSA 是以泛洪方式在 Area 0 中进行通告的，所以 Area 0 中的每台路由器都能够接收到所有其他路由器产生的 LSA。这样一来，不同路由器上的 LSDB 的内容其实是完全一样的。

稍微动动脑筋想想，我们便会发现，LSDB 其实完完全全地表征了 Area 0 中的各种信息，包括 Area 0 中共有多少台路由器，每台路由器有多少个接口，各个接口的类型和开销，路由器之间是怎样连接的，如此等等。也就是说，LSDB 相当于是一张关于 Area 0 的详细地图。

2. 最短路径树

图 8-25 中，每台路由器上的 LSDB 的内容都是完全一样。也就是说，每台路由器的“脑海”中都有一张相同的、关于整个 Area 0 的详细地图。显然，有了这张地图，每台路由器便可以从中找到从自己的位置去往地图中各个目的地的路线。然而，由于环路的存在，路由器是可以通过不同的路径（路线）从自己的位置去往同一个目的地的。在这种情况下，路由器必需根据路径开销的情况从不同的路径中选择出最优（即开销最小）的路径，这个过程也就是最短路径树（Shortest Path Tree，SPT）的生成过程。

图 8-25 中，每台路由器都会将 SPF 算法（Shortest Path First Algorithm）作用于自己“脑海”中的地图，从而生成一棵属于自己的 SPT。SPT 具有无环结构，其树根就是生成这棵 STP 的路由器，并且，从树根出发沿树干的指引去往某个目的地时，所经过的路径一定就是最优路径。

SPF 算法是由 Edsger W.Dijkstra 提出的，所以也称为 Dijkstra 算法。对于 SPF 算法的原理细节，我们这里不做描述。针对图 8-25 所示的网络，我们直接给出路由器 R1 的“脑海”中 SPT 的模样，以及路由器 R4 的“脑海”中 SPT 的模样，这些模样显示于图 8-26 中。

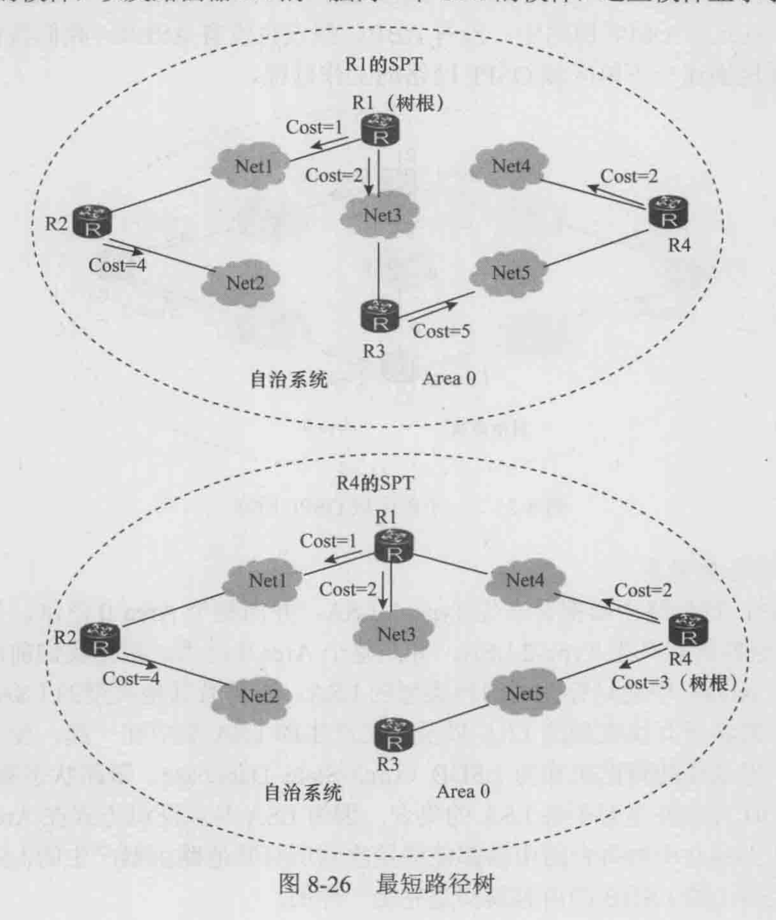


图 8-26 最短路径树

3. OSPF 路由表

OSPF 路由器在生成了自己的 SPT 后，便可以很容易地根据自己的 SPT 计算出从自己的位置去往各个目的地的路由。这些路由信息的集合，便是所谓的 OSPF 路由表。

我们将图 4-26 中 R1 的 SPT 进行细化，并重新显示在图 8-27 中。根据图 8-27，我们很容易知道，R1 的 OSPF 路由表中应该包含如下的路由信息。

（1）目的地：Net1。出接口：Intf-11。下一跳 IP 地址：Intf-11 的 IP 地址。Cost：1。
（2）目的地：Net2。出接口：Intf-11。下一跳 IP 地址：Intf-21 的 IP 地址。Cost：5。
（3）目的地：Net3。出接口：Intf-12。下一跳 IP 地址：Intf-12 的 IP 地址。Cost：2。
（4）目的地：Net4。出接口：Intf-12。下一跳 IP 地址：Intf-31 的 IP 地址。Cost：9。
（5）目的地：Net5。出接口：Intf-12。下一跳 IP 地址：Intf-31 的 IP 地址。Cost：7。
（6）……

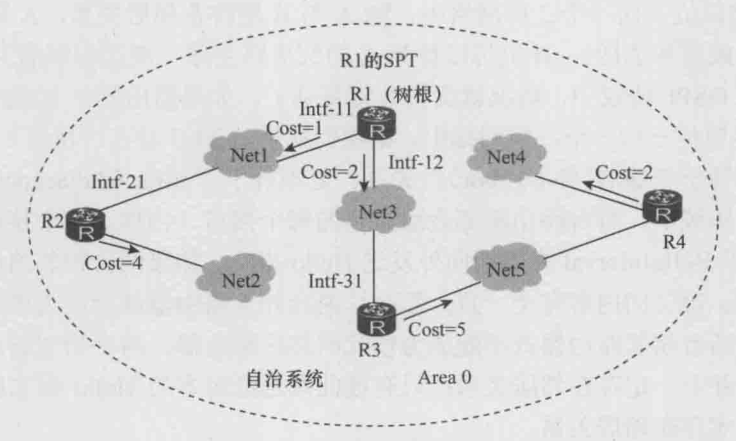


图 8-27 根据 SPT 进行路由计算

8.3.8 多区域 OSPF 网络

对于多区域 OSPF 网络的工作过程，我们这里只是给出非常粗略而简化的描述。

如图 8-22 所示，在多区域 OSPF 网络中，由于 ABR 和 ASBR 的存在，整个 OSPF 网络中除了有 Type-1 LSA 和 Type-2 LSA 之外，还有 Type-3、Type-4、Type-5 等类型的 LSA。也就是说，一台路由器的 LSDB 中，既有 Type-1 LSA 和 Type-2 LSA，也有其他类型的 LSA。根据自己的 LSDB 中的 Type-1 LSA 和 Type-2 LSA，路由器可以使用 SPF 算法得到自己的、关于本 Area 的 SPT，并根据 SPT 计算出自己去往本 Area 中各个目的地的路由（这个过程与单区域 OSPF 网络的工作过程完全一样）；根据自己的 LSDB 中的 Type-3 LSA，路由器可以使用 DV 算法计算出自己去往其他 Area 中各个目的地的路由；根据自己的 LSDB 中的 Type-4 LSA 和 Type-5 LSA，路由器可以使用 DV 算法计算出自己去往本 OSPF 网络（本自治系统）之外的目的地的路由。

例如，在图 8-22 中，R3 将 SPF 算法作用于自己的 LSDB 中的 Type-1 LSA 和 Type-2 LSA，便可得到自己的、关于 Area 1 的 SPT，并根据该 SPT 计算出自己去往 Area 1 中各个目的地的路由。另一方面，R3 将 DV 算法作用于自己的 LSDB 中的 Type-3 LSA，便可计算出自己去往 Area 0、Area 2、Area 3 中各个目的地的路由。同时，R3 将 DV 算法

作用于自己的 LSDB 中的 Type-4 LSA 和 Type-5 LSA，便可计算出自己去往整个 OSPF 网络之外的目的地的路由。

显然，多区域 OSPF 网络的工作过程要比单区域 OSPF 网络复杂得多。因此，我们自然会问，对于一个 OSPF 网络，什么情况应该采用单区域结构，什么情况下应该采用多区域结构？对于这个问题，希望读者朋友们自己去研究和思考。

8.3.9 邻居关系与邻接关系

在前面描述 RIP 协议的时候，我们曾提到过关于邻居路由器的概念。我们应该还记得，RIP 路由器都会每隔 30 秒钟向它所有的邻居路由器发布它的最新的 RIP 路由表中的所有路由信息，同时又不断地接收它的邻居路由器发来的路由信息，并根据这些接收到的路由信息来更新自己的 RIP 路由表。在 RIP 协议中，如果路由器 A 的某个接口和路由器 B 的某个接口位于同一个二层网络中，则 A 与 B 便存在邻居关系，A 可以称为 B 的邻居路由器（或简称邻居），B 也可以称为 A 的邻居路由器（或简称邻居）。

然而，在 OSPF 协议中，情况就变得非常复杂了。如果路由器 A 的某个接口和路由器 B 的某个接口位于同一个二层网络中，则我们就说 A 和 B 存在“相邻”关系，但“相邻”关系并不等于“邻居（Neighbor）”关系，更不等于“邻接（Adjacency）”关系。

在 OSPF 协议中，每台路由器都会通过它的每个接口（当然，这个接口必需使能了 OSPF 功能）以 HelloInterval 为周期向外发送 Hello 报文。如果两台相邻路由器彼此发送给对方的 Hello 报文的内容完全一致，那么这两台相邻路由器就会成为彼此的邻居路由器；否则，这两台相邻路由器就不能成为彼此的邻居路由器。两个相邻路由器之间存在相邻关系，但并不一定存在邻居关系；只有彼此发送给对方的 Hello 报文的内容完全一致，它们之间才存在邻居关系。

如果两台邻居路由器之间的二层网络类型是 P2P 网络或 P2MP 网络，则这两台邻居路由器一定会进入彼此之间的 LSDB 同步过程。当这两台邻居路由器成功地完成了 LSDB 同步之后，它们之间便建立起了邻接关系，也就是说，彼此成为了对方的邻接路由器。LSDB 同步过程的目的是要保证参与 LSDB 同步过程的两台邻居路由器最终能够拥有内容完全一致的 LSDB。LSDB 同步的过程是通过交互 OSPF DD 报文、OSPF LSR 报文、OSPF LSU 报文来实现的。关于 LSDB 同步的详细过程，我们这里不做描述。

如果两台邻居路由器之间的二层网络类型是 Broadcast 网络或 NBMA 网络，并且其中一台路由器是这个二层网络的 DR 或 BDR，那么这两台邻居路由器一定会进入彼此之间的 LSDB 同步过程。当这两台邻居路由器成功地完成了 LSDB 同步之后，它们之间便建立起了邻接关系。如果这两台邻居路由器都不是这个二层网络的 DR 或 BDR，那么，这两台邻居路由器就不会进入彼此之间的 LSDB 同步过程，也就是说，彼此之间是不可能建立起邻接关系的。

分清 OSPF 邻接关系和邻居关系是非常重要的。如果两台路由器之间存在邻接关系，则它们之间一定存在邻居关系。如果两台路由器之间存在邻居关系，则它们之间可能存在邻接关系，也可能不存在邻接关系。显然，一个 OSPF 网络中，邻接关系的数量总是等于或小于邻居关系的数量的。需要特别说明的是，在 OSPF 网络中，LSA 的泛洪过程只可能在具有邻接关系的路由器之间进行。LSA 的泛洪过程是通过交互 LSU 报文和 LSAck

报文而实现的（关于 LSA 泛洪的具体过程，我们这里不作描述）。显然，邻接关系的数量越少，网络中 OSPF 协议报文的数量就会越少，OSPF 协议占用的网络带宽资源以及路由器处理资源就会越少。

8.3.10 DR 与 BDR

在 P2P 网络或 P2MP 网络中，完全不存在 DR 与 BDR 的概念。DR 与 BDR 的概念只适用于 Broadcast 网络或 NBMA 网络。在 Broadcast 网络或 NBMA 网络中，DR 及 BDR 是通过选举（Election）而产生的。选举 DR 及 BDR 有两个目的，一个目的是让 DR 来产生针对这个 Broadcast 网络或 NBMA 网络的 Type-2 LSA，另一个目的是减少这个 Broadcast 网络或 NBMA 网络中邻接关系的数量。另外，BDR 的作用是：当 DR 出现故障时，BDR 能够迅速替代 DR 的角色。

在一个 Broadcast 网络或 NBMA 网络中，DR 会与所有其他的路由器（包括 BDR）建立邻接关系，BDR 也会与所有其他的路由器（包括 DR）建立邻接关系，除此之外，不能再有其他的邻接关系。

例如，图 8-28 所示的二层网络是一个以太网（注：以太网属于 Broadcast 网络类型），该网络包含了 6 台路由器和 1 台以太网交换机。在这个以太网中，如果任何两个邻居路由器之间都建立邻接关系，则总共会有 6×（6–1）÷2＝15 个邻接关系。

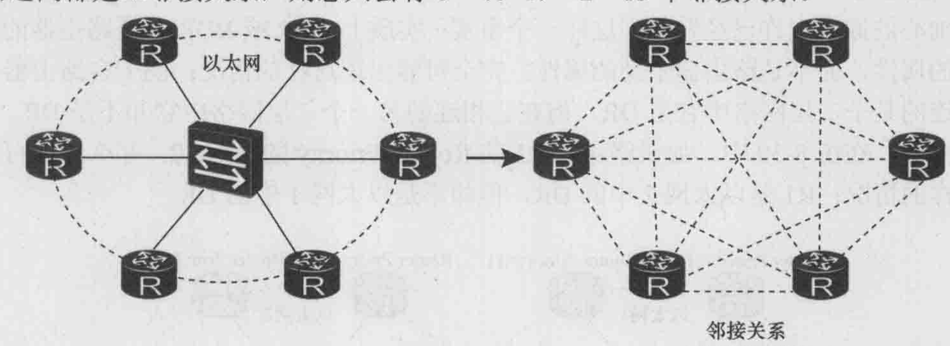


图 8-28 一个广播型网络

然而，在选举出 DR 和 BDR 之后，邻接关系的数量则会从原来的 15 个减少为 9 个，如图 8-29 所示。显然，如果该以太网中的路由器数量越多，则邻接关系数量减少的效果就越明显。

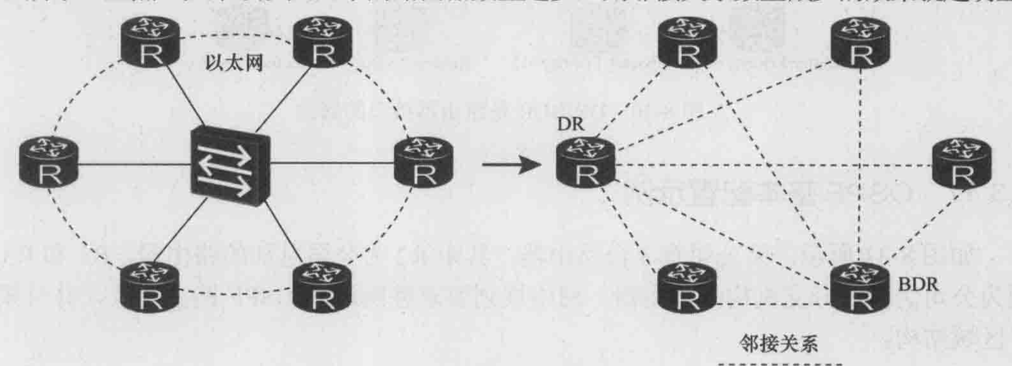


图 8-29 DR/BDR 可以减少邻接关系的数量

那么，DR 是如何被选举出来的呢？在一个 Broadcast 网络或 NBMA 网络中，路由器之间会进行 Hello 报文的交互，而每个 Hello 报文中总是携带了发送该 Hello 报文的路由器的 Router Priority 和 Router-ID。Router Priority 是一个 8bit 的二进制数，也可表示为十进制数，取值范围是从 0 到 255，并且取值越大，代表优先级越高。一个 Broadcast 网络或 NBMA 网络中的若干路由器在选举 DR 时，首先会比较各个路由器的 Router Priority 的值，Router Priority 的值最大者将被选举成为 DR；如果遇到 Router Priority 的值相等的情况，则 Router-ID 的值最大者将被选举成为 DR。注意，如果一个路由器的 Router Priority 的值为 0，则表明该路由器不会参加 DR 或 BDR 的选举过程。

如果一个 Broadcast 网络或 NBMA 网络中只存在 DR 而没有 BDR，那么当 DR 出现故障后，就需要重新选举 DR，而选举过程是需要耗费一定的时间的。如果网络中既有 DR，又有 BDR，则当 DR 出现故障后，BDR 就能迅速替代 DR 的角色。因此，BDR 的存在意义就是充当 DR 的备份，随时准备着迅速替代 DR 的角色。

BDR 的选举规则和过程与 DR 的选举规则和过程是完全一样的，但是需要注意的是，BDR 的选举是在选举出了 DR 之后进行的。选举 BDR 时，Router Priority 的值最大者将被选举成为 BDR。如果遇到 Router Priority 的值相等的情况，则 Router-ID 的值最大者将被选举为 BDR。另外需要注意的是，同一个 Broadcast 网络或 NBMA 网络中，BDR 和 DR 不能是同一路由器。

细心的读者也许已经发现了这样一个事实：实质上，DR 或 BDR 只是路由器的某个接口的属性，而不是路由器本身的属性。完全可能出现这样的情况：同一台路由器，在它相连的某个二层网络中它是 DR，但在它相连的另一个二层网络中它却不是 DR。

例如，在图 8-30 中，如果路由器 R1 的 Router Priority 的值为 10，那么完全可能出现这样的情况：R1 是以太网 2 中的 DR，但却不是以太网 1 中的 DR。

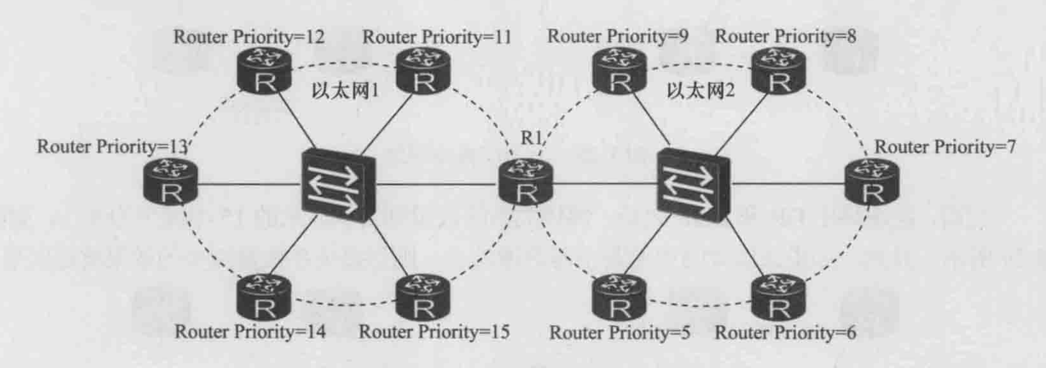


图 8-30 DR/BDR 是路由器接口的属性

8.3.11 OSPF 基本配置示例

如图 8-31 所示，某公司有 3 台路由器，其中 R2 为公司总部的路由器，R1 和 R3 分别为公司的两个分支机构的路由器。网络规划要求整网运行 OSPF 路由协议，并且采用多区域结构。

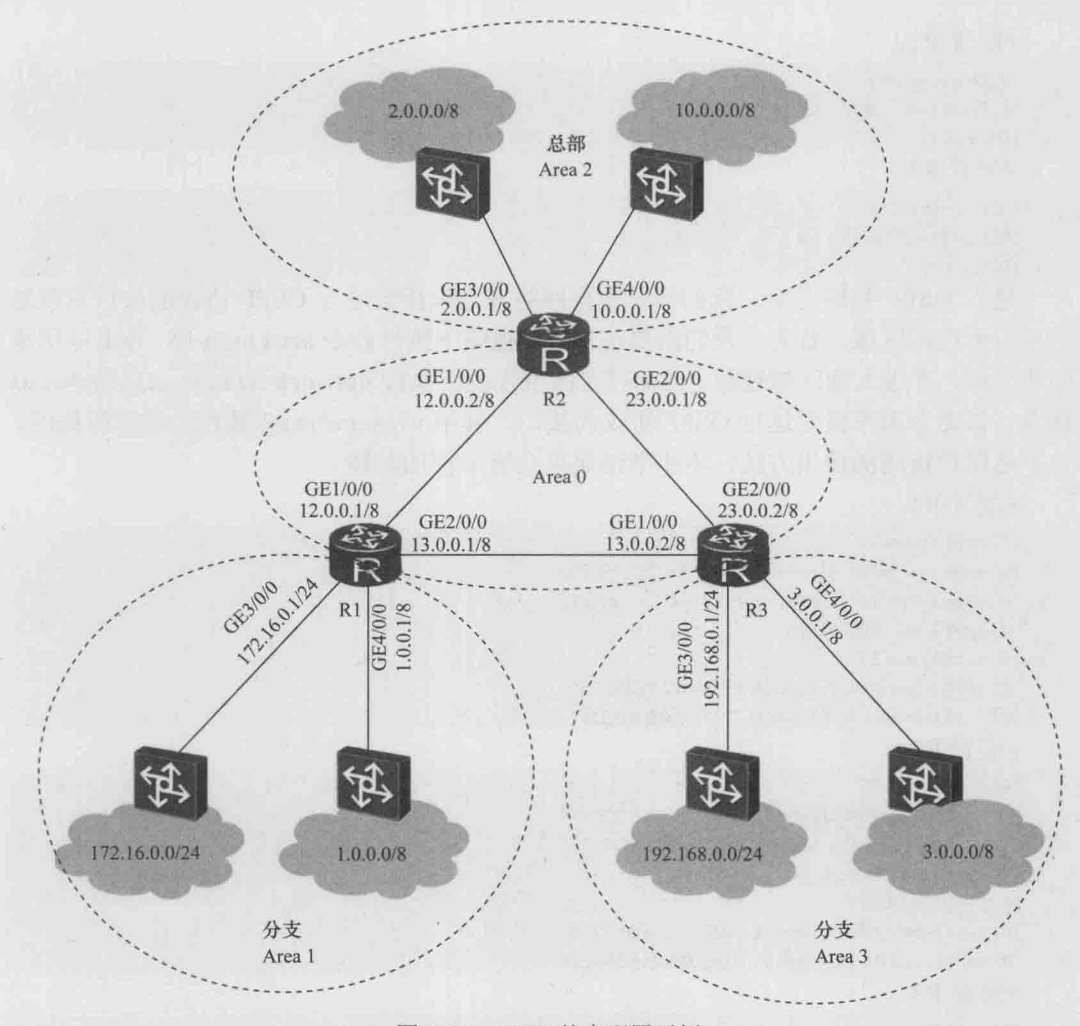


图 8-31　OSPF 基本配置示例

1. 配置思路

（1）在每台路由器上使能 OSPF 进程。

（2）根据区域的划分情况，指定各路由器接口的所属区域。

2. 配置步骤

要在路由器上配置 OSPF，必须首先进入系统视图，然后执行 **ospf** [*process-id* | **router-id** *router-id*]命令以使能 OSPF 进程，并进入 OSPF 视图。

执行 **ospf** 命令时，如果不输入 *process-id*（该参数表示 OSPF 进程编号）的值，则 *process-id* 默认取值为 1。*router-id* 是一个 32 比特的二进制数，也经常表示为点分十进制数。如果在执行 **ospf** 命令时不指定 *router-id*，则路由器会根据某种规则自动生成一个值来作为 *router-id*。

#配置 R1。

```
<R1> system-view
[R1] ospf   router-id 11.1.1.1   //在 R1 上使能 OSPF 进程，并且指定 R1 的 Router-ID 为 11.1.1.1
[R1-ospf-1]
```

#配置 R2。

```
<R2> system-view
[R2] ospf router-id 22.2.2.2
[R2-ospf-1]
```

#配置 R3。

```
<R3> system-view
[R3] ospf router-id 33.3.3.3
[R3-ospf-1]
```

进入 OSPF 视图之后，我们需要根据网络规划来指定运行 OSPF 协议的接口以及这些接口所在的区域。首先，我们需要在 OSPF 视图下执行命令 **area** *area-id*，该命令用来创建区域，并进入到区域视图。然后，在区域视图下执行 **network** *address wildcard-mask* 命令，该命令用来指定运行 OSPF 协议的接口，其中 *wildcard-mask* 被称为通配符掩码。关于通配符掩码的使用方法，本小节结尾处会有专门的解释。

#配置 R1。

```
[R1-ospf-1] area 0
[R1-ospf-1-area-0.0.0.0] network 12.0.0.0 0.255.255.255
[R1-ospf-1-area-0.0.0.0] network 13.0.0.0 0.255.255.255
[R1-ospf-1-area-0.0.0.0] quit
[R1-ospf-1] area 1
[R1-ospf-1-area-0.0.0.1] network 1.0.0.0 0.255.255.255
[R1-ospf-1-area-0.0.0.1] network 172.16.0.0 0.0.0.255
```

#配置 R2。

```
[R2-ospf-1] area 0
[R2-ospf-1-area-0.0.0.0] network 12.0.0.0 0.255.255.255
[R2-ospf-1-area-0.0.0.0] network 23.0.0.0 0.255.255.255
[R2-ospf-1-area-0.0.0.0] quit
[R2-ospf-1] area 2
[R2-ospf-1-area-0.0.0.2] network 2.0.0.0 0.255.255.255
[R2-ospf-1-area-0.0.0.2] network 10.0.0.0 0.255.255.255
```

#配置 R3。

```
[R3-ospf-1] area 0
[R3-ospf-1-area-0.0.0.0] network 13.0.0.0 0.255.255.255
[R3-ospf-1-area-0.0.0.0] network 23.0.0.0 0.255.255.255
[R3-ospf-1-area-0.0.0.0] quit
[R3-ospf-1] area 3
[R3-ospf-1-area-0.0.0.3] network 3.0.0.0 0.255.255.255
[R3-ospf-1-area-0.0.0.3] network 192.168.0.0 0.0.0.255
```

通过以上配置，各路由器之间应该都能建立起邻接关系。为了确认上述配置已经生效，我们可以使用 **display ospf** [*process-id*] **peer** 命令来查看路由器的邻居信息，以 R1 为例。

```
<R1> display ospf peer
                    OSPF Process 1 with Router ID 11.1.1.1
                                        Neighbors
 Area 0.0.0.0 interface 12.0.0.1（GigabitEthernet1/0/0）'s neighbors
 Router ID : 22.2.2.2          Address:    12.0.0.2
   State : Full   Mode : Nbr is   Master   Priority : 1
   DR : 12.0.0.2   BDR : 12.0.0.1   MTU : 0
 ......
                                                  Neighbors
```

```
Area 0.0.0.0 interface 13.0.0.1（GigabitEthernet2/0/0）'s neighbors
Router ID : 33.3.3.3        Address:   13.0.0.2
  State : Full    Mode :Nbr is    Master    Priority : 1
  DR : 13.0.0.2    BDR : 13.0.0.1    MTU : 0
……
```

回显信息中的第一个“State： Full”表明，R1 已经与 R2（Router-ID 为 22.2.2.2）成功建立了邻接关系，回显信息中的第二个“State： Full”表明，R1 已经与 R3（Router-ID 为 33.3.3.3）成功建立了邻接关系。

另外，回显信息中的“DR： 12.0.0.2 BDR： 12.0.0.1”表明，对于 R1 与 R2 之间的以太网，R2 被选举成为了 DR，R1 被选举成为了 BDR。回显信息中的“DR： 13.0.0.2 BDR：13.0.0.1”表明，对于 R1 与 R3 之间的以太网，R3 被选举成为了 DR，R1 被选举成为了 BDR。

display ospf [*process-id*] **routing** 命令可用来查看路由器的 OSPF 路由表，以 R1 为例。

```
[R1] display ospf routing
                          OSPF Process 1 with Router ID 11.1.1.1
                                   Routing Tables
 Routing for Network
 Destination        Cost   Type          NextHop        AdvRouter      Area
 1.0.0.0/8          1      Stub          1.0.0.1        11.1.1.1       0.0.0.1
 12.0.0.0/8         1      Transit       12.0.0.1       11.1.1.1       0.0.0.0
 13.0.0.0/8         1      Transit       13.0.0.1       11.1.1.1       0.0.0.0
 172.16.0.0/24      1      Stub          172.16.0.1     11.1.1.1       0.0.0.1
 2.0.0.0/8          2      Inter-area    12.0.0.2       22.2.2.2       0.0.0.0
 3.0.0.0/8          2      Inter-area    13.0.0.2       33.3.3.3       0.0.0.0
 10.0.0.0/8         2      Inter-area    12.0.0.2       22.2.2.2       0.0.0.0
 23.0.0.0/8         2      Transit       13.0.0.2       33.3.3.3       0.0.0.0
 23.0.0.0/8         2      Transit       12.0.0.2       22.3.3.3       0.0.0.0
 192.168.0.0/24     2      Inter-area    13.0.0.2       33.3.3.3       0.0.0.0
 Total Nets : 10
 Intra Area : 6    Inter Area:   4    ASE : 0    NSSA : 0
```

可以看到，R1 的 OSPF 路由表中已经拥有了从 R1 去往各个目的网络的路由。

现在，我们来解释一下命令 **network** *address wildcard-mask* 中通配符掩码的使用方法。命令 **network** *address wildcard-mask* 中，*address* 是一个 32bit 的二进制数，也可以表示为一个点分十进制数；*wildcard-mask* 是一个通配符掩码，也是一个 32bit 的二进制数，并且也可以表示为一个点分十进制数。*wildcard-mask* 与 *address* 合写在一起时，表示的是一个由若干个 IP 地址组成的集合，这个集合中的任何一个 IP 地址都满足且只需满足这样的条件：如果 *wildcard-mask* 中的某一个比特位的取值为 0，则该 IP 地址中的对应比特位的取值必须与 *address* 中的对应比特位的取值相同。

例如，如果 *address* 为 12.0.0.0，*wildcard-mask* 为 0.0.0.0，则它们所表示的 IP 地址集合中只有唯一的 1 个 IP 地址，这个 IP 地址就是 12.0.0.0。如果 *address* 为 12.0.0.0，*wildcard-mask* 为 8.0.0.1，则它们所表示的 IP 地址集合中共有 4 个 IP 地址，这 4 个 IP 地址分别是：12.0.0.0、12.0.0.1、4.0.0.0、4.0.0.1。如果 *address* 为 12.0.0.0，*wildcard-mask* 为 0.255.255.255，则它们所表示的 IP 地址集合中包含了范围在 12.0.0.0~12.255.255.255 的所有 16 777 216 个 IP 地址。

接下来，我们来看一个配置命令，内容如下。

```
[R5-ospf-4-area-0.0.0.3] network address wildcard-mask
```

该配置命令的含义是：如果 R5 的某个接口的 IP 地址属于 *address* 和 *wildcard-mask* 所表示的 IP 地址集合，那么该接口就需要在 Area 3 中参与进程编号为 4 的 OSPF 进程。

再看一个配置命令，代码如下。

```
[R1-ospf-1-area-0.0.0.0] network 12.0.0.0 0.255.255.255
```

该配置命令的含义是：如果 R1 的某个接口的 IP 地址属于 12.0.0.0~12.255.255.255 这个范围，那么该接口就需要在 Area 0 中参与进程编号为 1 的 OSPF 进程。细心的读者可能已经发现，这个配置命令其实就是取自前面的配置示例（见图 8-31）。从图 8-31 中我们可以看到，R1 的 GE1/0/0 接口的 IP 地址为 12.0.0.1，该地址是属于 12.0.0.0~12.255.255.255 这个范围的，所以这个配置命令的作用其实就是让 R1 的 GE1/0/0 接口在 Area 0 中参与进程编号为 1 的 OSPF 进程。

8.3.12 练习题

1.（单选）从原理性角度看，OSPF 与 RIP 的主要差别是？（ ）

A．RIP 是一种慢收敛的路由协议，而 OSPF 是一种快收敛的路由协议

B．RIP 是一种基于 DV 算法的路由协议，而 OSPF 是一种基于链路状态的路由协议

C．RIP 只能以跳数作为路由开销的定义，而 OSPF 则没有这个限制

D．RIP 只能适用于规模较小的网络，而 OSPF 则没有这个限制

2.（多选）关于 OSPF 的区域化结构，下列描述中正确的是？（ ）

A．一个多区域 OSPF 网络中，骨干区域只能有一个

B．一个多区域 OSPF 网络中，至少存在一个 ABR

C．一个 ABR 肯定也是一个骨干路由器

D．一个骨干路由器肯定也是一个 ABR

E．ASBR 只能出现在骨干区域中

F．一个多区域 OSPF 网络中有且只能有一个 ASBR

G．非骨干区域之间的通信必需通过骨干区域中转才能实现

3.（多选）在 OSPF 协议中，需要选举 DR/BDR 的网络类型有？（ ）

A．Broadcast 网络

B．NBMA（Non-Broadcast Multi-Access）网络

C．P2P 网络

D．P2MP 网络

4.（单选）OSPF 协议报文的类型一共有几种？（ ）

A．3 种　　B．4 种　　C．5 种　　D．6 种

5.（多选）关于 LSA（Link-State Advertisement），下列描述中正确的是？（ ）

A．一个 OSPF Hello 报文中至少携带了一条完整的 LSA

B．LSA 是一种 OSPF 协议报文

C．在一个 OSPF 网络中，如果两个链路状态数据库彼此之间实现了同步，那么这两个链路状态数据库中所包含的 Type-1 LSA 的条数一定是相同的

D．在一个多区域 OSPF 网络中，Router LSA 的泛洪是可以跨区域的

E．在一个多区域 OSPF 网络中，Network LSA 的泛洪是可以跨区域的

F．在一个多区域 OSPF 网络中，Network Summary LSA 的泛洪是可以跨区域的

G．Network Summary LSA 是由 ASBR 产生的

H．ASBR Summare LSA 是由 ABR 产生的

6.（多选）关于 OSPF 邻居关系和邻接关系，下列描述中正确的是？（　）

A．在一个 OSPF 网络中，邻居关系的数量与邻接关系的数量有可能是相等的

B．对于一个 OSPF 网络中的某个路由器而言，它的邻居路由器的数量总是等于或小于它的邻接路由器的数量

C．同一广播网络中的 DR 和 BDR 之间应该建立起邻接关系

D．在一个多区域 OSPF 网络中，ABR 是不可能与任何其他的路由器建立邻接关系的

7.（多选）关于 DR/BDR 的选举问题，下列描述中正确的是？（　）

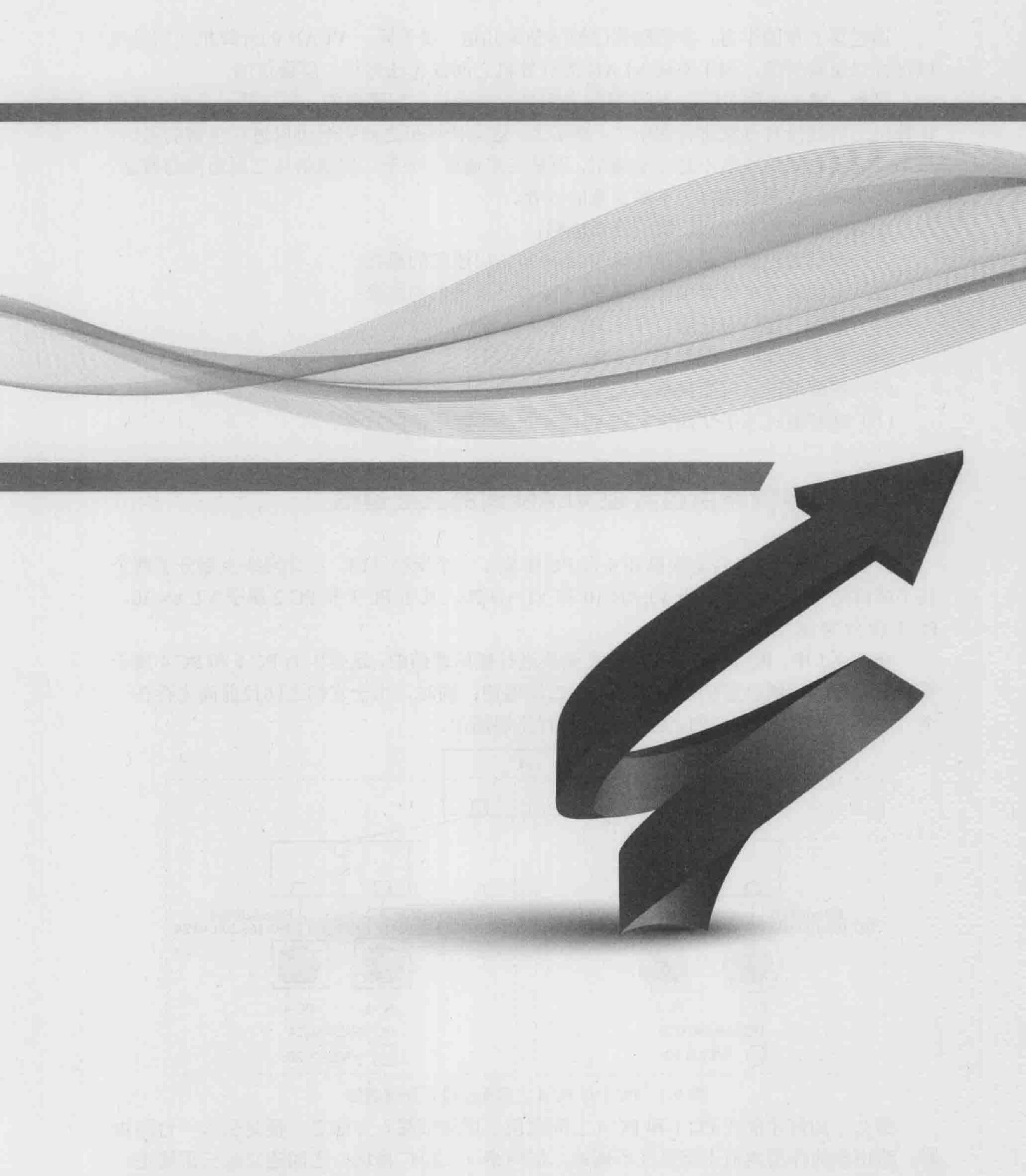
A．如果一个 Broadcast 网络中某台路由器的 Router Priority 的值是 255，那么这台路由器一定会被选举成为该 Broadcast 网络的 DR

B．如果一个 Broadcast 网络中某台路由器的 Router Priority 的值是 0，那么这台路由器一定会被选举成为该 Broadcast 网络的 DR

C．同一个 Broadcast 网络中的 DR 和 BDR，它们的 Router Priority 的值绝对不可能相等

D．以上选项都是错误的

第9章 VLAN间的三层通信



通过第 5 章的学习，我们应该已经清楚地知道，属于同一 VLAN 的计算机之间是可以进行二层通信的，属于不同 VLAN 的计算机之间是无法进行二层通信的。

虽然，属于不同 VLAN 的计算机之间是无法进行二层通信的，但这并不是说，这些计算机之间就没有办法进行通信了。事实上，这些计算机之间完全可以进行正常的通信，只不过它们之间的通信不是二层通信，而是三层通信。关于二层通信与三层通信的概念，请读者朋友们认真去复习一下第 5 章的内容。

学习完本章内容之后，我们应该能够：

（1）理解通过多臂路由器实现 VLAN 间三层通信的原理；

（2）理解通过单臂路由器实现 VLAN 间三层通信的原理；

（3）理解二层口与三层口在行为特征上的差异；

（4）从原理性的角度理解什么是三层交换机；

（5）理解通过三层交换机实现 VLAN 内的二层通信的原理；

（6）理解通过三层交换机实现 VLAN 间的三层通信的原理。

9.1 通过多臂路由器实现 VLAN 间的三层通信

如图 9-1 所示，3 台交换机和 4 台 PC 组成了一个交换网络，在此网络上划分了两个基于端口的 VLAN，分别为 VLAN 10 和 VLAN 20，其中 PC 1 和 PC 2 属于 VLAN 10，PC 3 和 PC 4 属于 VLAN 20。

在图 9-1 中，PC 1 与 PC 4 之间是无法进行任何通信的，这是因为 PC 1 和 PC 4 属于不同的 VLAN，所以它们之间无法进行二层通信；同时，由于它们之间目前尚未存在一个“三层通道”，所以它们之间也无法进行三层通信。

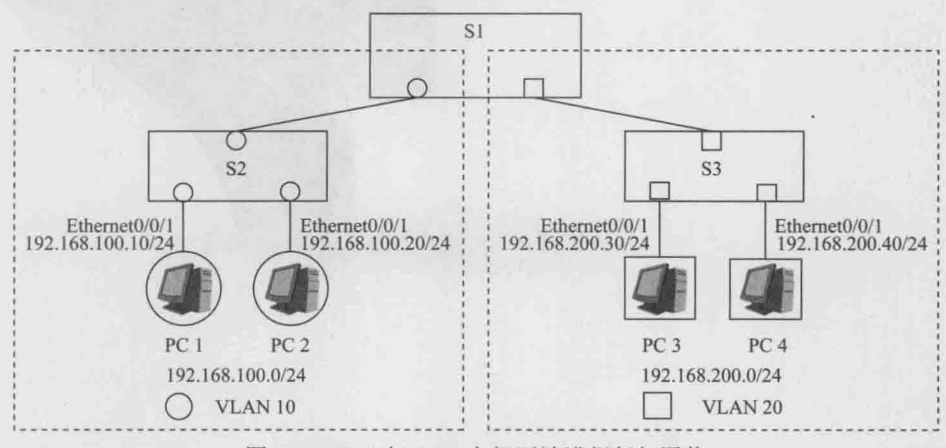


图 9-1 PC 1 与 PC 4 之间无法进行任何通信

那么，如何才能在 PC 1 和 PC 4 之间实现三层通信呢？方法之一便是引入一台路由器。路由器的作用实质上就是在不同的二层网络（二层广播域）之间建立起三层通道。不同的 VLAN 其实就是不同的二层网络（二层广播域），所以路由器当然也可以在不同的 VLAN 之间建立起三层通道。

在图 9-1 所示的网络中引入一台路由器 R，便得到了图 9-2 所示的网络。从图 9-2 中

我们看到，路由器 R 的 GE1/0/0 接口与交换机 S1 的属于 VLAN 10 的 D1 端口相连，路由器 R 的 GE2/0/0 接口与交换机 S1 的属于 VLAN 20 的 D2 端口相连。需要特别提醒读者的是，与PC的接口一样，路由器R的GE1/0/0接口和GE2/0/0接口都是不能发送和接收Tagged VLAN 帧的。另外，从图 9-2 中我们也看到，路由器 R 分别从 GE1/0/0 接口和 GE2/0/0 接口各自引出了一条物理链路，每条物理链路可以被形象地称为路由器的一条“手臂”，所以这里的路由器 R 也常常被形象地称为“双臂路由器”，或泛泛地称为“多臂路由器”。

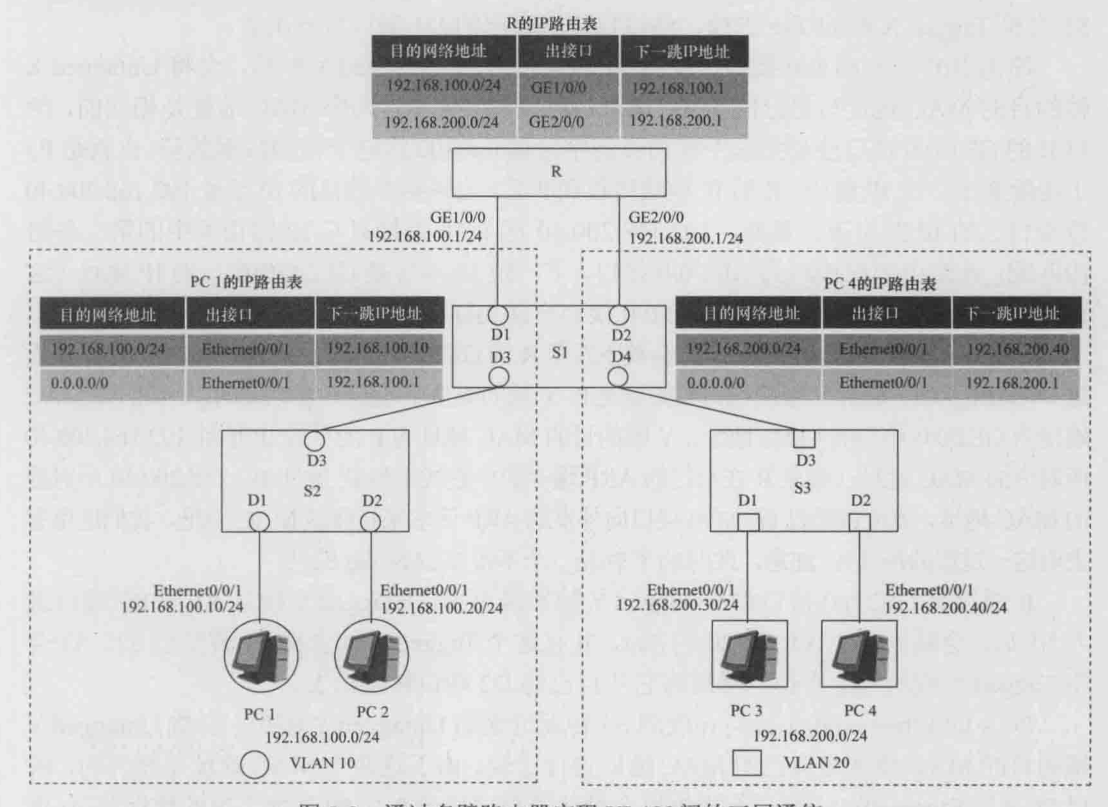


图 9-2　通过多臂路由器实现 VLAN 间的三层通信

接下来，我们通过一个例子来说明 PC 1 和 PC 4 是如何实现三层通信的，也就是说，PC 1 是如何将一个名为 P 的 IP 报文成功地发送给 PC 4 的。图 9-2 中，交换机的 Access 端口有：S2 的 D1 端口和 D2 端口，S3 的 D1 端口和 D2 端口，S1 的 D1 端口和 D2 端口。交换机的 Trunk 端口有：S2 的 D3 端口，S3 的 D3 端口，S1 的 D3 端口和 D4 端口。

首先，P 是在 PC 1 的网络层形成的，P 的目的 IP 地址为 192.168.200.40，源 IP 地址为 192.168.100.10。然后，根据 P 的目的 IP 地址，PC 1 会进行 IP 路由表的查询工作（图 9-2 中展示了 R、PC 1、PC 4 的简化后的 IP 路由表）。PC 1 的 IP 路由表中有两条路由，其中一条为缺省路由。显然，P 的目的 IP 地址 192.168.200.40 只能匹配上那条缺省路由，该路由的出接口为 PC 1 的 Ethernet0/0/1 接口，下一跳 IP 地址为路由器 R 的 GE1/0/0 接口的 IP 地址 192.168.100.1（路由器 R 的 GE1/0/0 接口也因此被称为是 192.168.100.0/24 或 VLAN 10 的缺省网关）。

于是，根据这条缺省路由的指示，P 会被下发至 PC 1 的 Ethernet0/0/1 接口，并被封装成一个帧。假设这个帧取名为 X，那么 X 帧的载荷数据就是 P，X 帧的类型字段的值为

0x0800，X 帧的源 MAC 地址为 PC 1 的 Ethernet0/0/1 接口的 MAC 地址，X 帧的目的 MAC 地址为路由器 R 的 GE1/0/0 接口的 MAC 地址（如果 PC 1 在自己的 ARP 缓存表中查找不到 IP 地址 192.168.100.1 所对应的 MAC 地址，就应该通过 ARP 机制去获取该 MAC 地址，我们这里省去对这一过程的描述）。注意，此时的 X 帧是一个不带 VLAN Tag 的帧。

接下来，PC 1 会从 Ethernet0/0/1 接口将 Untagged X 帧发送出去。X 帧从 S2 的 D1 端口进入 S2 后，会被添加上 VLAN 10 的 Tag，并且这个 Tagged X 帧会被 S2 转发至 S1。S1 会将 Tagged X 帧的 Tag 去掉，然后将它从自己的 D1 端口转发出去。

路由器 R 的 GE1/0/0 接口在收到 S1 转发过来的 Untagged X 帧后，会将 Untagged X 帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 地址是相同的，所以 R 的 GE1/0/0 接口会根据这个帧的类型字段值 0x0800 将这个帧的数据载荷（也就是 P）上送给 R 的三层 IP 模块。R 的 IP 模块接收到 P 后，会根据 P 的目的 IP 地址 192.168.200.40 查询自己的 IP 路由表。显然，192.168.200.40 这个 IP 地址只与 IP 路由表中的第二条路由匹配，该路由的出接口为 GE2/0/0 接口，下一跳 IP 地址是 GE2/0/0 接口的 IP 地址（这说明 P 要去往的目的网络是与 GE2/0/0 接口直接相连的）。

于是，根据这条路由的指示，P 会被下发至 R 的 GE2/0/0 接口，并被封装成一个帧。假设这个帧取名为 Y，那么 Y 帧的载荷数据就是 P，Y 帧的类型字段的值为 0x0800，Y 帧的源 MAC 地址为 GE2/0/0 接口的 MAC 地址，Y 帧的目的 MAC 地址为 P 的目的 IP 地址 192.168.200.40 所对应的 MAC 地址（如果 R 在自己的 ARP 缓存表中查找不到 IP 地址 192.168.200.40 所对应的 MAC 地址，就应该通过 GE2/0/0 接口向外发送 ARP 请求来获取该 MAC 地址，我们这里省去对这一过程的描述）。注意，此时的 Y 帧是一个不带 VLAN Tag 的帧。

R 通过其 GE2/0/0 接口将 Untagged Y 帧发送出去。Untagged Y 帧从 S1 的 D2 端口进入 S1 后，会被添加上 VLAN 20 的 Tag，并且这个 Tagged Y 帧会被 S1 转发至 S3。S3 会将 Tagged Y 帧的 Tag 去掉，然后将它从自己的 D2 端口转发出去。

PC 4 的 Ethernet0/0/1 接口在收到 S3 转发过来的 Untagged Y 帧后，会将 Untagged Y 帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 地址是相同的，所以 PC 4 的 Ethernet0/0/1 接口会根据这个帧的类型字段值 0x0800 将这个帧的数据载荷（也就是 P）上送给 PC 4 的位于三层的 IP 模块。

至此，源于 PC 1 的三层 IP 模块的 IP 报文 P 便成功地到达了 PC 4 的三层 IP 模块，属于 VLAN 10 的 PC 1 与属于 VLAN 20 的 PC 4 之间成功地进行了一次三层通信。

细心的读者可能已经发现，图 9-2 所示的网络中，路由器 R 与交换机 S1 之间存在一个物理环路。针对这个物理环路，请读者朋友们思考这样一个问题：假设这个网络没有划分 VLAN，同时也假设所有的交换机都没有运行 STP（Spanning Tree Protocol），那么，当 PC 1 发送出一个广播帧后，这个广播帧会因为 R 与 S1 之间的物理环路而导致广播风暴的产生吗？正确答案应该是不会产生广播风暴。

9.2 通过单臂路由器实现 VLAN 间的三层通信

VLAN 间的三层通信可以通过多臂路由器来实现，但这种实现方法面临的一个主要

问题是：每一个 VLAN 都需要占用路由器上的一个物理接口（也就是说，每一个 VLAN 都需要路由器从一个物理接口伸出一只手臂来），如果 VLAN 数目众多，就需要占用大量的路由器接口。事实上，路由器的物理接口资源是非常宝贵而稀缺的，一台路由器上的物理接口数量通常都是非常有限的，无法支持数量较多的 VLAN。实际的网络部署中，几乎都不会通过多臂路由器来实现 VLAN 间的三层通信。

为了节省路由器的物理接口资源，我们还可以通过采用单臂路由器的方法来实现 VLAN 间的三层通信。采用这种方法时，必须对路由器的物理接口进行“子接口（Sub-Interface）”划分。一个路由器的物理接口可以划分为多个子接口，不同的子接口对应了不同的 VLAN。这些子接口的 MAC 地址均为“衍生”出它们的那个物理接口的 MAC 地址，但是它们的 IP 地址各不相同。一个子接口的 IP 地址应该配置为该子接口所对应的那个 VLAN 的缺省网关地址。子接口是一个逻辑上的概念，所以子接口也常常被称为虚接口。

如图 9-3 所示，路由器 R 的物理接口 GE1/0/0 被划分成了两个子接口，分别为 GE1/0/0.1 和 GE1/0/0.2。GE1/0/0.1 对应了 VLAN 10，GE1/0/0.2 对应了 VLAN 20。GE1/0/0.1 的 IP 地址为 192.168.100.1/24，也就是 VLAN 10 的缺省网关地址；GE1/0/0.2 的 IP 地址为 192.168.200.1/24，也就是 VLAN 20 的缺省网关地址。子接口 GE1/0/0.1 的 MAC 地址和 GE1/0/0.2 的 MAC 地址是一样的，都是物理接口 GE1/0/0 的 MAC 地址。

注意，在图 9-3 中，交换机的 Access 端口有：S2 的 D1 端口和 D2 端口，S3 的 D1 端口和 D2 端口。交换机的 Trunk 端口有：S2 的 D3 端口，S3 的 D3 端口，S1 的 D3 端口、D2 端口和 D1 端口。属于 VLAN 10 的帧和属于 VLAN 20 的帧都需要被允许通过 S1 的 D1 端口。S1 与 R 之间的链路是一个 VLAN Trunk 链路，该链路上运动的帧必须是带有 VLAN Tag 的。这也意味着，子接口 GE1/0/0.1 或 GE1/0/0.2 向外发送的帧也必须是带有 VLAN Tag 的。

接下来，我们还是通过一个例子来说明图 9-3 中的 PC 1 和 PC 4 之间是如何实现三层通信的，也就是说，PC 1 是如何将一个名为 P 的 IP 报文成功地发送给 PC 4 的。

首先，P 是在 PC 1 的网络层形成的，P 的目的 IP 地址为 192.168.200.40，源 IP 地址为 192.168.100.10。然后，根据 P 的目的 IP 地址，PC 1 会进行 IP 路由表的查询工作。PC 1 的 IP 路由表中有两条路由，其中一条为缺省路由。显然，P 的目的 IP 地址 192.168.200.40 只能匹配上那条缺省路由，该路由的出接口为 PC 1 的 Ethernet0/0/1 接口，下一跳 IP 地址为路由器 R 的 GE1/0/0.1 子接口的 IP 地址 192.168.100.1（路由器 R 的 GE1/0/0.1 子接口也因此被称为是 192.168.100.0/24 或 VLAN 10 的缺省网关）。

于是，根据这条缺省路由的指示，P 会被下发至 PC 1 的 Ethernet0/0/1 接口，并被封装成一个帧。假设这个帧取名为 X，那么 X 帧的载荷数据就是 P，X 帧的类型字段的值为 0x0800，X 帧的源 MAC 地址为 PC 1 的 Ethernet0/0/1 接口的 MAC 地址，X 帧的目的 MAC 地址为路由器 R 的 GE1/0/0.1 子接口的 MAC 地址（如果 PC 1 在自己的 ARP 缓存表中查找不到 IP 地址 192.168.100.1 所对应的 MAC 地址，就应该通过 ARP 机制去获取该 MAC 地址，我们这里省去对这一过程的描述）。注意，此时的 X 帧是一个不带 VLAN Tag 的帧。

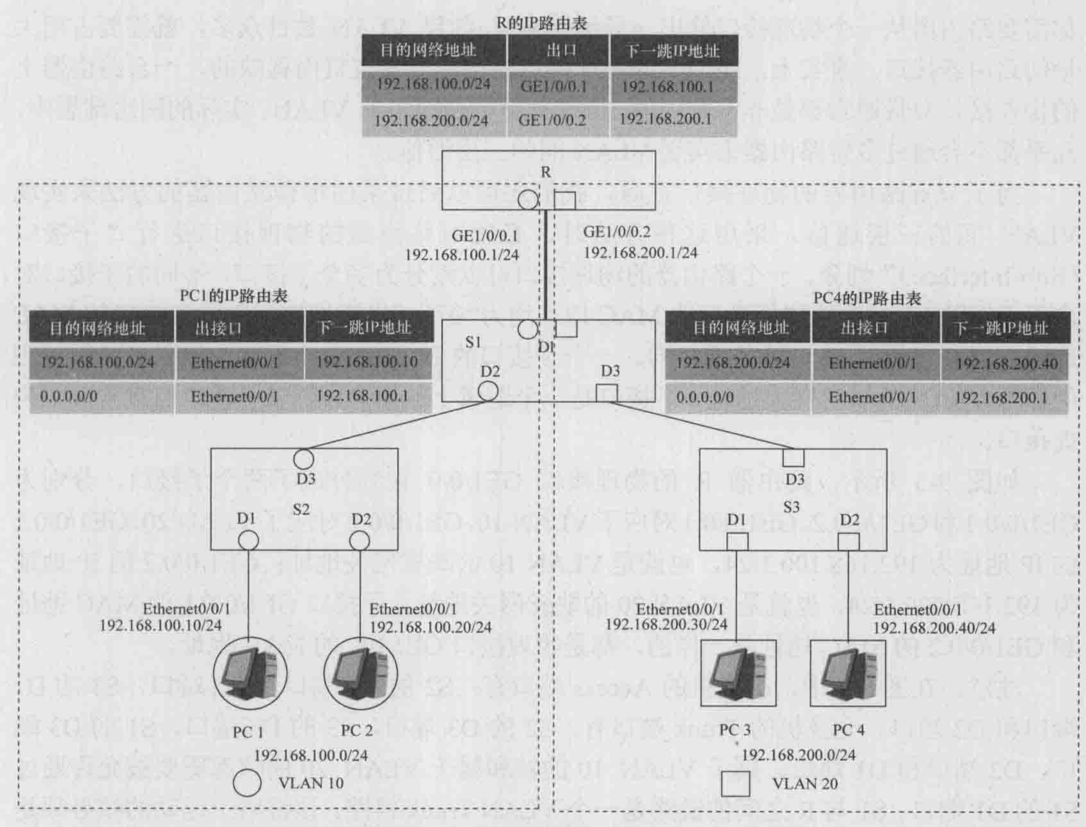


图 9-3　通过单臂路由器实现 VLAN 间的三层通信

接下来，PC 1 会从 Ethernet0/0/1 接口将 Untagged X 帧发送出去。Untagged X 帧从 S2 的 D1 端口进入 S2 后，会被添加上 VLAN 10 的 Tag，并且这个 Tagged X 帧会被 S2 和 S1 转发至路由器的物理接口 GE1/0/0。

路由器 R 的物理接口 GE1/0/0 在收到 S1 转发过来的 Tagged X 帧后，发现这个帧是属于 VLAN 10 的，于是这个帧会被交给子接口 GE1/0/0.1 来处理。子接口 GE1/0/0.1 发现，Tagged X 帧的目的 MAC 地址正是自己的 MAC 地址，并且这个帧的类型字段的值是 0x0800，于是子接口 GE1/0/0.1 会将这个帧的载荷数据（也就是 P）上送给路由器 R 的三层 IP 模块。

路由器 R 的 IP 模块接收到 P 后，会根据 P 的目的 IP 地址 192.168.200.40 查询自己的 IP 路由表。显然，192.168.200.40 这个 IP 地址只与 IP 路由表中的第二条路由匹配，该路由的出接口为子接口 GE1/0/0.2，下一跳 IP 地址是子接口 GE1/0/0.2 的 IP 地址（这说明 P 要去往的目的网络是与子接口 GE1/0/0.2 直接相连的）。

于是，根据这条路由的指示，P 会被下发至 R 的 GE1/0/0.2 子接口，并被封装成一个帧。假设这个帧取名为 Y，那么 Y 帧的载荷数据就是 P，Y 帧的类型字段的值为 0x0800，Y 帧的源 MAC 地址为子接口 GE1/0/0.2 的 MAC 地址，Y 帧的目的 MAC 地址为 P 的目的 IP 地址 192.168.200.40 所对应的 MAC 地址（如果 R 在自己的 ARP 缓存表中查找不到 IP 地址 192.168.200.40 所对应的 MAC 地址，就应该通过子接口 GE1/0/0.2 向外发送

ARP 请求来获取该 MAC 地址，我们这里省去对这一过程的描述）。注意，Y 帧还必须带上 VLAN 20 的 Tag！

路由器 R 将 Tagged Y 帧从其子接口 GE1/0/0.2 发送出去之后（从物理直观上讲，就是从 GE1/0/0 这个物理接口发送出去），该 Tagged Y 帧会到达交换机 S3 的 D2 端口。然后，S3 会将 Tagged Y 帧的 Tag 去掉，然后将它从自己的 D2 端口转发出去。

PC 4 的 Ethernet0/0/1 接口在收到 S3 转发过来的 Untagged Y 帧后，会将 Untagged Y 帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 地址是相同的，所以 PC 4 的 Ethernet0/0/1 接口会根据这个帧的类型字段值 0x0800 将这个帧的数据载荷（也就是 P）上送给 PC 4 的位于三层的 IP 模块。

至此，源于 PC 1 的三层 IP 模块的 IP 报文 P 便成功地到达了 PC 4 的三层 IP 模块，属于 VLAN 10 的 PC 1 与属于 VLAN 20 的 PC 4 之间成功地进行了一次三层通信。

9.3　通过三层交换机实现 VLAN 间的三层通信

VLAN 间的三层通信可以通过多臂路由器或单臂路由器来实现。通过单臂路由器来实现时，可以节约路由器的物理接口资源，但是，这种方式也有其不足之处。如果 VLAN 的数量众多，VLAN 间的通信流量很大时，单臂链路所能提供的带宽就有可能无法支撑这些通信流量。另外，如果单臂链路一旦发生了中断，那么所有 VLAN 间的通信也都会因此而中断。为此，人们引入了一种被称为“三层交换机”的网络设备，并通过三层交换机来更经济、更快速、更可靠地实现 VLAN 间的三层通信。在说明什么是三层交换机之前，我们必须先解释一下关于“二层口”和“三层口”的概念。

平时，我们通常会混用“端口”和“接口”这两个词，端口也就是接口，接口也就是端口，它们都是“网口”的意思。本书的习惯是，交换机上的网口称端口，路由器或计算机上的网口称接口。但是，这个习惯并不重要，只是一个习惯而已，读者不必太在意。那么，什么是二层口呢？什么又是三层口呢？

通常，我们把交换机上的端口称为二层端口，或简称为二层口；同时，我们把路由器或计算机上的接口称为三层接口，或简称为三层口。二层口的行为特征与三层口的行为特征存在明显的差异，具体如下。

（1）二层口只有 MAC 地址，没有 IP 地址；三层口既有 MAC 地址，又有 IP 地址。

（2）设备的某个二层口在接收到一个广播帧后，会将这个广播帧从该设备的其他所有二层口泛洪出去。

（3）设备的某个三层口在接收到一个广播帧后，会根据这个广播帧的类型字段的值将这个广播帧的载荷数据上送到该设备第三层的相应模块去处理。

（4）设备的某个二层口在接收到一个单播帧后，该设备会在自己的 MAC 地址表中查找这个帧的目的 MAC 地址。如果查不到这个 MAC 地址，则该设备会将这个帧从其他所有二层口泛洪出去。如果查到了这个 MAC 地址，则比较 MAC 地址表项所指示的那个二层口是不是这个帧进入该设备时所通过的那个二层口。如果是，则设备会将这个

帧直接丢弃；如果不是，则设备会把这个帧从 MAC 地址表项所指示的那个二层口转发出去。

（5）设备的某个三层口在接收到一个单播帧后，会比较这个帧的目的 MAC 地址是不是该三层口的 MAC 地址。如果不是，则会直接将这个帧丢弃；如果是，则根据这个帧的类型字段的值将这个帧的载荷数据上送到该设备第三层的相应模块去处理。

（6）关于设备的二层口或三层口接收到一个组播帧的情况，本书不进行分析和描述。

上面几点就是对二层口的行为特征和三层口的行为特征的总结性描述。二层口的行为特征与三层口的行为特征的差异，直接引出了交换机与路由器的差异。

① 交换机的端口都是二层口，一台交换机的不同二层口之间只存在二层转发通道，不存在三层转发通道。交换机内部存在 MAC 地址表，用以进行二层转发。交换机内部不存在 IP 路由表。

② 路由器的端口都是三层口，一台路由器的不同三层口之间只存在三层转发通道，不存在二层转发通道。路由器内部存在 IP 路由表，用以进行三层转发。路由器内部不存在 MAC 地址表。

现在，我们可以来解释什么是三层交换机了。三层交换机的原理性定义是：三层交换机是二层交换机与路由器的一种集成形式，它除了可以拥有一些二层口外，还可以拥有一些“混合端口”（简称为“混合口”）。混合口既具有二层口的行为特征，同时又具有三层口的行为特征。一台三层交换机上，不同的混合口之间同时存在二层转发通道和三层转发通道，不同的二层口之间只存在二层转发通道，一个混合口与一个二层口之间也只存在二层转发通道。三层交换机内既存在 MAC 地址表，用以进行二层转发，又存在 IP 路由表，用以进行三层转发。一台三层交换机上可以只有混合口，而无二层口。一台三层交换机上也可以只有二层口，而无混合口（此时的三层交换机完全退化成了一台二层交换机）。

接下来，我们就通过几个例子来说明三层交换机是如何实现 VLAN 内的二层通信以及 VLAN 间的三层通信的。

如图 9-4 所示，PC 1 和 PC 3 被划分进了 VLAN 10，PC 2 和 PC 4 被划分进了 VLAN 20。S1 是一台三层交换机，S2 和 S3 都是二层交换机。为了能够支持 VLAN 10 与 VLAN 20 之间的三层通信，我们需要在 S1 上配置两个逻辑意义上的 VLAN 接口，这两个 VLAN 接口分别称为 VLANIF 10 和 VLANIF 20（注：VLANIF 中的 IF 是 Interface 的缩写）。VLANIF 10 和 VLANIF 20 具有三层口的行为特征，并拥有自己的 IP 地址。我们把 VLANIF 10 和 VLANIF 20 的 IP 地址分别配置为 192.168.100.1/24 和 192.168.200.1/24，这两个 IP 地址其实分别就是 VLAN 10 和 VLAN 20 的缺省网关地址。这样一来，S1 上的端口 GE1/0/0 和端口 GE2/0/0 就都成了混合口，这两个混合口一方面具有二层口的行为特征，同时又具有三层口的行为特征。

注意，在图 9-4 中，交换机的 Access 端口有：S2 的 D1 端口和 D2 端口，S3 的 D1 端口和 D2 端口。交换机的 Trunk 端口有：S2 的 D3 端口，S3 的 D3 端口，S1 的 GE1/0/0 端口和 GE2/0/0 端口。

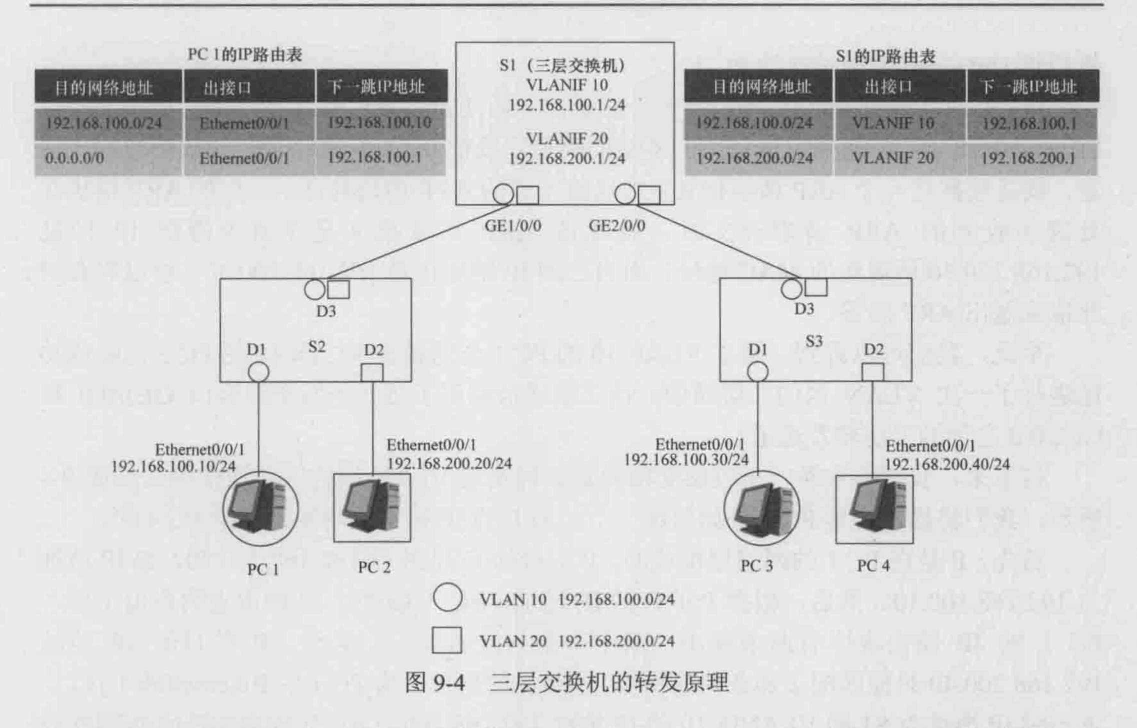


图 9-4 三层交换机的转发原理

现在，我们先来看一下三层交换机是如何实现同一 VLAN 内的二层通信的。如图 9-4 所示，假设 PC 1 需要发送一个 ARP 请求，去询问 PC 3 的 MAC 地址是多少，也就是询问 IP 地址 192.168.100.30 所对应的 MAC 地址是多少。我们需要描述清楚这个 ARP 请求是如何到达 PC 3 的。

首先，PC 1 的数据链路层（二层）需要准备好一个广播帧，假设这个广播帧取名为 X。X 帧的目的 MAC 地址为 ff-ff-ff-ff-ff-ff，源 MAC 地址为 PC 1 的 Ethernet0/0/1 接口的 MAC 地址，X 帧的类型字段的值为 0x0806，X 帧的载荷数据是一个 ARP 请求报文，该 ARP 请求报文的作用是请求得到 IP 地址 192.168.100.30 所对应的 MAC 地址。注意，X 帧现在是不带 VLAN Tag 的。

PC 1 通过其 Ethernet0/0/1 接口发送出 Untagged X 帧后，Untagged X 帧会从 S2 的 D1 端口进入 S2，并被 S2 添加上 VLAN 10 的 Tag。然后，Tagged X 帧会到达 S1 的混合端口 GE1/0/0。

因为 S1 的 GE1/0/0 端口具有三层口的行为特征，并且 GE1/0/0 端口收到的 Tagged X 帧是一个广播帧，所以，GE1/0/0 端口会根据 Tagged X 帧的类型字段值 0x0806 将 Tagged X 帧的载荷数据（注意，载荷数据是一个 ARP 请求报文）上送给三层的 ARP 模块处理。三层的 ARP 模块在处理所收到的 ARP 请求报文时，发现该 ARP 报文是在请求得到 IP 地址 192.168.100.30 所对应的 MAC 地址，而自己的 IP 地址（也就是 VLANIF 10 的 IP 地址）是 192.168.100.1，所以不会做出 ARP 应答，而是直接将这个 ARP 请求报文丢弃。

同时，因为 S1 的 GE1/0/0 端口又具有二层口的行为特征，并且 GE1/0/0 端口接收到的 Tagged X 帧是一个广播帧，所以，GE1/0/0 端口会将这个 Tagged X 帧从 GE2/0/0 端口泛洪出去。

然后，Tagged X 帧会运动到 S3 的 D1 端口。S3 的 D1 端口会去掉 Tagged X 帧的 Tag，

然后将 Untagged X 帧发送给 PC 3。

PC 3 的 Ethernet0/0/1 接口是一个三层口，而 Untagged X 又是一个广播帧，所以 Ethernet0/0/1 接口会根据 Untagged X 帧的类型字段值 0x0806 将这个帧的载荷数据（注意，载荷数据是一个 ARP 请求报文）上送给三层的 ARP 模块处理。三层的 ARP 模块在处理所收到的 ARP 请求报文时，发现该 ARP 请求报文是在请求得到 IP 地址 192.168.100.30 所对应的 MAC 地址，而自己的 IP 地址正是 192.168.100.30，所以将会对此请求做出 ARP 应答。

至此，我们可以看到，属于 VLAN 10 的 PC 1 与同属于 VLAN 10 的 PC 3 已经成功地进行了一次 VLAN 内的二层通信。该二层通信利用了 S1 的两个混合口 GE1/0/0 和 GE2/0/0 之间的二层转发通道。

接下来，我们再来看一下三层交换机是如何实现 VLAN 间的三层通信的。如图 9-4 所示，我们将描述清楚 PC 1 是如何将一个名为 P 的 IP 报文成功地发送给 PC 4 的。

首先，P 是在 PC 1 的网络层形成的，P 的目的 IP 地址为 192.168.200.40，源 IP 地址为 192.168.100.10。然后，根据 P 的目的 IP 地址，PC 1 会进行 IP 路由表的查询工作。PC 1 的 IP 路由表中有两条路由，其中一条为缺省路由。显然，P 的目的 IP 地址 192.168.200.40 只能匹配上那条缺省路由，该路由的出接口为 PC 1 的 Ethernet0/0/1 接口，下一跳 IP 地址为 S1 的 VLANIF 10 的 IP 地址 192.168.100.1（S1 的 VLANIF 10 也因此被称为是 192.168.100.0/24 或 VLAN 10 的缺省网关）。

于是，根据这条缺省路由的指示，P 会被下发至 PC 1 的 Ethernet0/0/1 接口，并被封装成一个单播帧。假设这个帧取名为 X，那么 X 帧的载荷数据就是 P，X 帧的类型字段的值为 0x0800，X 帧的源 MAC 地址为 PC 1 的 Ethernet0/0/1 接口的 MAC 地址，X 帧的目的 MAC 地址为 VLANIF 10 的 IP 地址所对应的 MAC 地址（如果 PC 1 在自己的 ARP 缓存表中查找不到 IP 地址 192.168.100.1 所对应的 MAC 地址，就应该通过 ARP 机制去获取该 MAC 地址，我们这里省去对这一过程的描述）。注意，此时的 X 帧是一个不带 VLAN Tag 的帧。

然后，PC 1 会从 Ethernet0/0/1 接口将 Untagged X 帧发送出去。Untagged X 帧从 S2 的 D1 端口进入 S2 后，会被添加上 VLAN 10 的 Tag，并且这个 Tagged X 帧会被送达至 S1 的 GE1/0/0 端口。

因为 S1 的 GE1/0/0 端口具有三层口的行为特征，并且 GE1/0/0 端口收到的 Tagged X 帧是一个单播帧，所以，GE1/0/0 端口会将这个帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 是相同的，所以 GE1/0/0 端口会根据这个帧的类型字段值 0x0800 将载荷数据（也就是 P）上送给三层 IP 模块进行处理。

S1 的 IP 模块接收到 P 后，会根据 P 的目的 IP 地址 192.168.200.40 查询自己的 IP 路由表。显然，192.168.200.40 这个 IP 地址只与 IP 路由表中的第二条路由匹配，该路由的出接口为 VLANIF 20，下一跳 IP 地址是 VLANIF 20 的 IP 地址（这说明 P 要去往的目的网络是与 VLANIF 20 直接相连的，但目前还不清楚究竟是与 GE1/0/0 端口直接相连，还是与 GE2/0/0 端口直接相连）。

于是，根据这条路由的指示，P 会被下发至 VLANIF 20 接口（但目前还不清楚，究竟应该下发至 GE1/0/0 端口，还是应该下发至 GE2/0/0 端口），并被封装成一个单播帧。

假设这个帧取名为 Y，那么 Y 帧的载荷数据就是 P，Y 帧的类型字段的值为 0x0800，Y 帧的目的 MAC 地址应该是 P 的目的 IP 地址 192.168.200.40 所对应的 MAC 地址，但目前还不清楚 Y 帧的源 MAC 地址应该是 GE1/0/0 端口的 MAC 地址还是 GE2/0/0 端口的 MAC 地址。

假设 S1 现在还不知道 192.168.200.40 所对应的 MAC 地址，那么 S1 就需要通过其 GE1/0/0 端口向外发送 ARP 广播请求去获取这个 MAC 地址，同时 S1 还需要通过其 GE2/0/0 端口向外发送 ARP 广播请求去获取这个 MAC 地址。注意，从 GE1/0/0 端口向外发送的 ARP 广播帧以及从 GE2/0/0 端口向外发送的 ARP 广播帧都必须带上 VLAN 20 的 Tag。显然，最后的结果是，GE2/0/0 端口会收到 ARP 应答（从而学习到了 192.168.200.40 所对应的 MAC 地址），GE1/0/0 端口不会收到 ARP 应答。这样一来， Y 帧的源 MAC 地址现在就可以确定为是 GE2/0/0 端口的 MAC 地址，Y 帧的出端口应该为 GE2/0/0，而不是 GE1/0/0。另外，Y 帧的目的 MAC 地址就是通过 ARP 机制而学习到的 MAC 地址。注意，Y 帧还必须带上 VLAN 20 的 Tag。

然后，S1 将 Tagged Y 帧从 GE2/0/0 端口发送出去，该 Tagged Y 帧会到达交换机 S3 的 D2 端口。然后，S3 会将 Tagged Y 帧的 Tag 去掉，然后将它从自己的 D2 端口转发出去。

PC 4 的 Ethernet0/0/1 接口在收到 S3 转发过来的 Untagged Y 帧后，会将 Untagged Y 帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 地址是相同的，所以 PC 4 的 Ethernet0/0/1 接口会根据这个帧的类型字段值 0x0800 将这个帧的数据载荷（也就是 P）上送给 PC 4 的位于三层的 IP 模块。

至此，源于 PC 1 的三层 IP 模块的 IP 报文 P 便成功地到达了 PC 4 的三层 IP 模块，属于 VLAN 10 的 PC 1 与属于 VLAN 20 的 PC 4 之间成功地进行了一次三层通信。

然而，我们不要忘记了这样一个细节。当初，Tagged X 帧被送达至 S1 的 GE1/0/0 端口后，因为 S1 的 GE1/0/0 端口具有二层口的行为特征，所以 GE1/0/0 端口还应该以二层口的行为特征来对 Tagged X 帧进行处理。处理的方式有两种，分别说明如下。

方式一：Tagged X 帧到达 GE1/0/0 后，GE1/0/0 的三层口发现 Tagged X 帧是一个单播帧，于是会将 Tagged X 帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 地址是相同的，所以 GE1/0/0 的三层口会将 Tagged X 帧的载荷数据 P 上送给三层 IP 模块进行后续处理，同时通知 GE1/0/0 的二层口不要对 Tagged X 帧进行任何处理。也就是说，针对 Tagged X 帧，GE1/0/0 的二层口的行为特征受到了抑制。

方式二：Tagged X 帧到达 GE1/0/0 后，GE1/0/0 的三层口发现 Tagged X 帧是一个单播帧，于是会将 Tagged X 帧的目的 MAC 地址与自己的 MAC 地址进行比较。由于这两个 MAC 地址是相同的，所以 GE1/0/0 的三层口会将 Tagged X 帧的载荷数据 P 上送给三层 IP 模块进行后续处理，但是不会通知 GE1/0/0 的二层口不要对 Tagged X 帧进行处理。于是，S1 会去自己的 MAC 地址表中查找 Tagged X 帧的目的 MAC 地址。显然，在 MAC 地址表中是查不到 Tagged X 帧的目的 MAC 地址的，于是 S1 会将 Tagged X 帧从 GE2/0/0 端口泛洪出去。被泛洪的 Tagged X 帧会被送达至 S3 的 D1 端口（注意，Tagged X 帧不会被送达至 S3 的 D2 端口，因为 Tagged X 帧带有 VLAN 10 的 Tag，而 D2 端口是属于 VLAN 20 的），D1 端口会去掉 Tagged X 帧的 Tag，然后将 Untagged X 帧发送给 PC 3 的 Ethernet0/0/1 接口。PC 3 的 Ethernet0/0/1 接口会将自己的 MAC 地址与 Untagged X 帧的

目的 MAC 地址进行比较。由于这两个 MAC 地址不相同，所以 PC 3 的 Ethernet0/0/1 接口会将接收到的 Untagged X 帧直接丢弃。

至此，我们便完整地描述了如何利用三层交换机来实现 VLAN 内的二层通信以及 VLAN 间的三层通信。

读者朋友们可能经常听到另外一些关于三层交换机的说法，诸如三层交换机比路由器便宜一些，三层交换机比路由器的转发速度快一些，三层交换机是“一次路由，多次转发”，而路由器是“一次路由，一次转发”，如此等等。需要说明的是，这些说法中所隐含的道理已经超出了本书的知识范围，所以我们这里不做分析。

9.4 VLANIF 接口配置示例

如图 9-5 所示，启用 S1 的三层交换功能，并通过在三层交换机 S1 上配置 VLANIF 接口，实现不同 VLAN 间用户的三层通信。

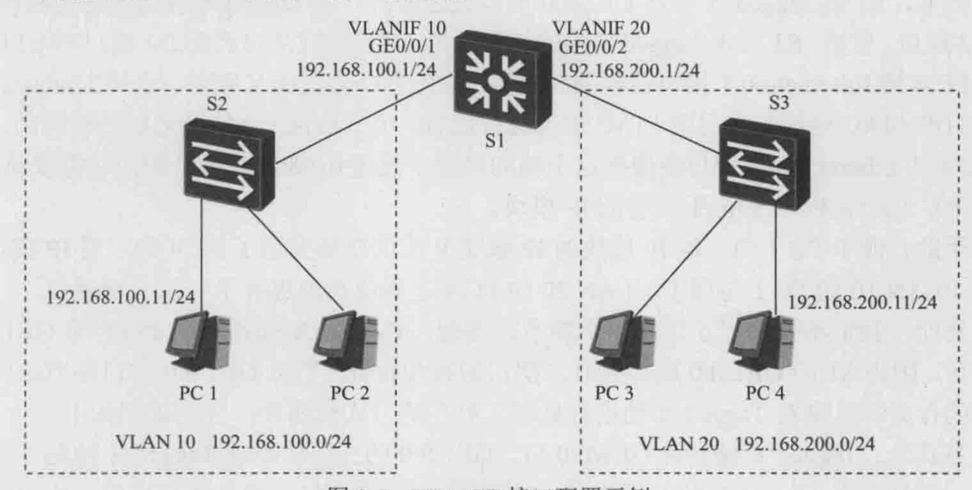


图 9-5 VLANIF 接口配置示例

1. 配置思路

（1）在交换机上 S1 创建 VLAN（注意，在 S2 和 S3 上无需创建 VLAN）。

（2）配置交换机 S1 的端口。

（3）在交换机 S1 上创建 VLANIF 接口并配置 IP 地址，实现不同 VLAN 之间的三层互通。

2. 配置步骤

S1 上创建 VLAN 10 和 VLAN 20。

```
<S1> system-view
[S1] vlan batch 10 20
```

S1 上进行端口配置。

```
[S1] interface gigabitethernet 0/0/1
[S1-GigabitEthernet0/0/1] port link-type access
[S1-GigabitEthernet0/0/1] port default vlan 10
[S1-GigabitEthernet0/0/1] quit
[S1] interface gigabitethernet 0/0/2
```

```
[S1-GigabitEthernet0/0/2] port link-type access
[S1-GigabitEthernet0/0/2] port default vlan 20
[S1-GigabitEthernet0/0/2] quit
```

现在，我们对所做的配置进行确认，使用的命令是 **display vlan** *vlan-id* **verbose**。我们以 S1 上的 VLAN 10 为例。

```
<S1> display vlan 10 verbose
* : Management-VLAN
----------------------------
  VLAN ID                                : 10
  ......
  VLAN State                             : Up
  -------------------------
  Untagged      Port : GigabitEthernet0/0/1
  -------------------------
  Active Untag    Port : GigabitEthernet0/0/1
----------------------------
Interface                                Physical
GigabitEthernet0/0/1                     UP
......
```

从回显信息中我们可以看到，S1 上 VLAN10 内已经加入了端口 GigabitEthernet0/0/1。

接下来，我们使用命令 **display port vlan** 查看 S1 上所有 VLAN 所包含的端口信息。

```
<S1> display port vlan
Port                  Link Type       PVID    Trunk VLAN List
--------------------------------------------------------------------------
GigabitEthernet0/0/1   access          10       -
GigabitEthernet0/0/2   access          20       -
......
```

从回显信息中我们可以看到，S1 的 GigabitEthernet0/0/1 和 GigabitEthernet0/0/2 已经配置成为了 access 类型的端口，并且 PVID 的值也是正确的。这说明我们所配置的命令已经在设备上生效了。

接下来，我们需要在 S1 上创建 VLANIF 接口并配置 IP 地址。执行命令 **interface vlanif** *vlan-id* 可以创建 VLANIF 接口，并进入 VLANIF 接口视图，然后执行命令 **ip address** *ip-address* { *mask* | *mask-length* }，为 VLANIF 接口配置 IP 地址。

S1 上配置 VLANIF 接口。

```
[S1] interface vlanif 10
[S1-Vlanif10] ip address 192.168.100.1 24
[S1-Vlanif10] quit
[S1] interface vlanif 20
[S1-Vlanif20] ip address 192.168.200.1 24
[S1-Vlanif20] quit
```

现在，我们对配置好的 VLANIF 接口进行确认，使用的命令是 **display ip interface brief vlanif** *vlan-id*。我们以 VLANIF 10 接口为例。

```
<S1> display ip interface brief vlanif 10
*down:   administratively down
!down:   FIB overload down
^down:   standby
(l):   loopback
(s):   spoofing
Interface                              IP Address/Mask         Physical     Protocol
```

```
Vlanif10                                192.168.100.1/24        up          up
```

从回显信息中我们看到，VLANIF 10 的接口状态和链路层协议状态都已经是 UP，并且该 VLANIF 接口已经配置了 IP 地址，说明该 VLANIF 接口已经配置成功。

以上就是需要在 S1 上进行的所有配置。接下来，我们可以配置 PC 1 和 PC 4 的 IP 地址和缺省网关地址，然后，通过在 PC 1 上验证能否 Ping 通 PC 4，来检验我们配置的三层交换机能否实现不同 VLAN 间用户的三层通信。

将 PC 1 的 IP 地址配置为 192.168.100.11/24，并将其缺省网关地址配置为 S1 上 VLANIF 10 的 IP 地址 192.168.100.1/24。将 PC 4 的 IP 地址配置为 192.168.200.11/24，并将其缺省网关地址配置为 S1 上 VLANIF 20 的 IP 地址 192.168.200.1/24。

配置完成后，在 PC 1 上执行命令“ping 192.168.200.11”。

```
C:\>ping 192.168.200.11

Ping 192.168.200.11 : 32 data bytes,    Press Ctrl_C to break
From 192.168.200.11 : bytes=32 seq=1 ttl=127 time=62 ms
From 192.168.200.11 : bytes=32 seq=2 ttl=127 time=47 ms
From 192.168.200.11 : bytes=32 seq=3 ttl=127 time=62 ms
From 192.168.200.11 : bytes=32 seq=4 ttl=127 time=63 ms
From 192.168.200.11 : bytes=32 seq=5 ttl=127 time=62 ms

--- 192.168.200.11 ping statistics ---
  5 packet（s）transmitted
  5 packet（s）received
  0.00% packet loss
  round-trip min/avg/max = 47/59/63 ms
```

从回显信息中我们可以看到，PC 1 收到了 PC 4 的响应，表示 PC 1 可以 Ping 通 PC 4，这说明三层交换机 S1 成功地实现了 VLAN 10 与 VLAN 20 之间的三层通信。

9.5 练习题

1.（多选）通过以下哪些设备可以实现不同 VLAN 间的通信？（　）

A. 路由器　　B. 二层交换机　　C. 三层交换机

2.（多选）下列描述中正确的是？（　）

A. 路由器内部可以存在 MAC 地址表

B. 路由器内部不存在 MAC 地址表

C. 三层交换机内部可以存在路由表

D. 三层交换机内部可以存在 MAC 地址表

3.（多选）下列描述中正确的是？（　）

A. 路由器的子接口需要配置 IP 地址

B. 路由器的子接口无需配置 IP 地址

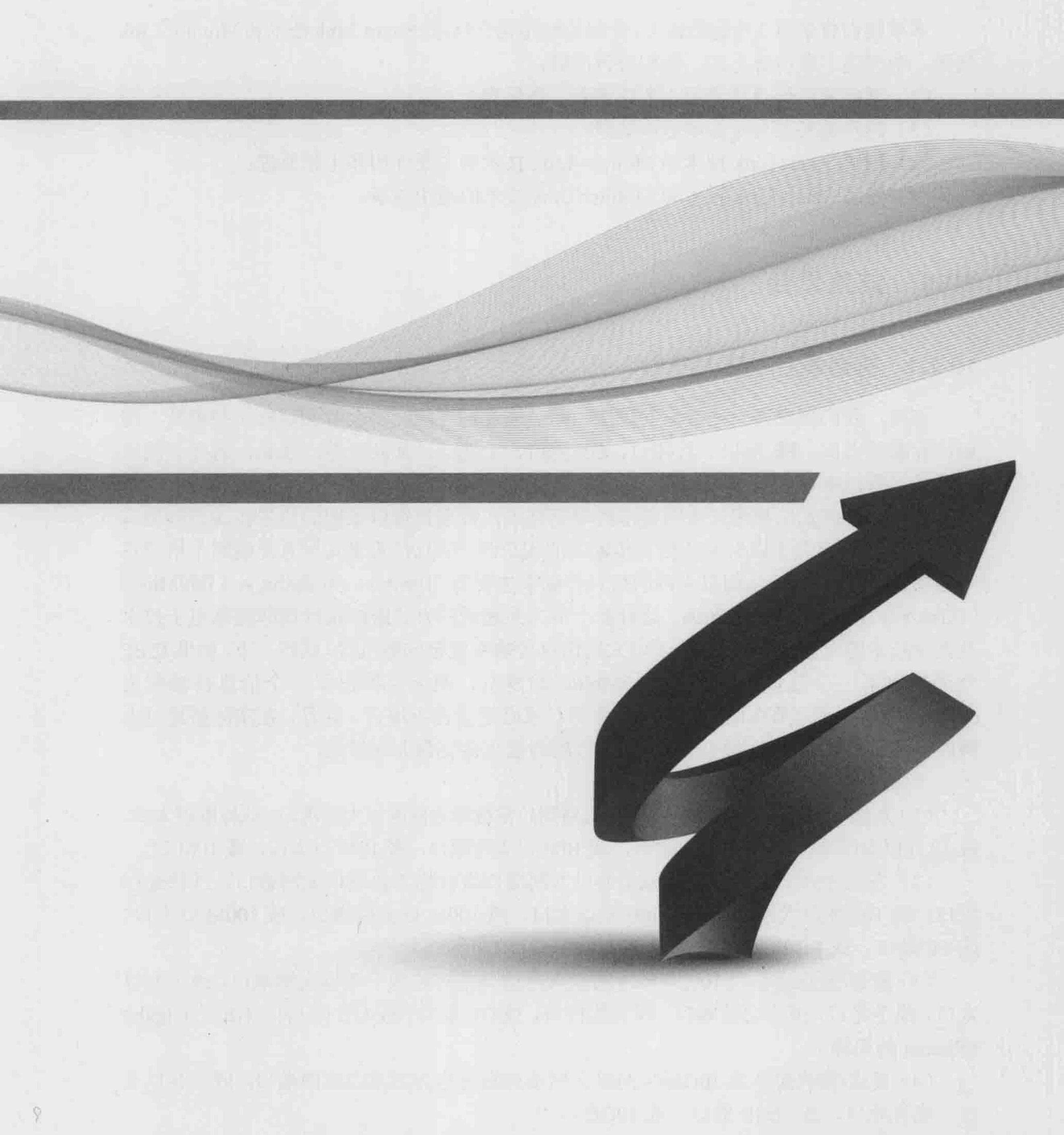
C. 三层交换机的 VLANIF 接口需要配置 IP 地址

D. 三层交换机的 VLANIF 接口无需配置 IP 地址

4.（多选）下列描述中正确的是？（　）

A．路由器上不同的三层口之间可以建立二层转发通道
B．路由器上不同的三层口之间可以建立三层转发通道
C．二层交换机上不同的二层口之间可以建立二层转发通道
D．二层交换机上不同的二层口之间可以建立三层转发通道
E．三层交换机上不同的二层口之间可以建立二层转发通道
F．三层交换机上不同的混合口之间可以建立三层转发通道
G．三层交换机上不同的混合口之间可以建立二层转发通道

第10章 链路技术



本章我们将学习3种链路技术，分别是链路聚合技术、Smart Link 技术和 Monitor Link 技术。学习完本章内容之后，我们应该能够：

（1）理解链路聚合技术的主要作用和工作原理；

（2）熟悉链路聚合技术的适用场景；

（3）理解 Smart Link 技术和 Monitor Link 技术的主要作用和工作原理；

（4）熟悉 Smart Link 技术和 Monitor Link 技术的适用场景。

10.1 链路聚合

10.1.1 链路聚合的基本概念

首先，我们来澄清一些常见的说法。读者朋友们可能经常会听到这样一些说法，例如：标准以太口、FE 端口、百兆口、GE 端口、千兆口，如此等等。那么，这些说法究竟是什么意思呢？

其实，这些说法都跟以太网技术的规范有关，特别是跟以太网的信息传输率规范有关。IEEE 在制定关于以太网的信息传输率的规范时，信息传输率几乎总是按照十倍关系来递增的。目前，规范化的以太网的信息传输率主要有：10Mbit/s，100Mbit/s，1 000Mbit/s（1Gbit/s），10Gbit/s，100Gbit/s。这种按十倍关系递增的方式既能很好地匹配微电子技术及光学技术的发展，又能控制关于以太网信息传输率规范的散乱性。试想一下，如果 IEEE 今天推出了一个信息传输率为 415Mbit/s 的规范，明天又推出了一个信息传输率为 624Mbit/s 的规范，那么以太网网卡的生产厂家必定会苦不堪言。并且，在实际搭建以太网的时候，以太网链路两端的端口速率匹配问题也会变得非常散乱。

下面是对一些常见说法的澄清。

（1）发送/接收速率为 10Mbit/s 的以太网端口常被称为标准以太网端口，或标准以太口，或 10 兆以太网端口，或 10 兆以太口，或 10M 以太网端口，或 10M 以太口，或 10M 口。

（2）发送/接收速率为 100Mbit/s 的以太网端口常被称为快速以太网端口，或快速以太口，或 100 兆以太网端口，或 100 兆以太口，或 100M 以太网端口，或 100M 以太口，或 FE 端口，或 FE 口（注：FE 是 Fast Ethernet 的简称）。

（3）发送/接收速率为 1000Mbit/s 的以太网端口常被称为千兆以太网端口，或千兆以太口，或千兆口，或吉比特端口，或吉比特口，或 GE 端口，或 GE 口（注：GE 是 Gigabit Ethernet 的简称）。

（4）发送/接收速率为 10Gbit/s 的以太网端口常被称为万兆以太网端口，或万兆以太口，或万兆口，或 10GE 端口，或 10GE 口。

（5）发送/接收速率为 100Gbit/s 的以太网端口常被称为 100GE 端口，或 100GE 口。

以太网链路的说法是与以太网端口的说法相对应的。例如，如果一条链路两端的端口是 GE 口，则这条链路就称为一条 GE 链路；如果一条链路两端的端口是 FE 口，则这条链路就称为一条 FE 链路，如此等等。

现在说说什么是链路聚合技术。图 10-1 示意了某个公司的网络结构，交换机 S1 接

入了 10 个用户，每个用户都通过一条 FE 链路与 S1 相连，S1 与核心交换机 S2 之间的链路是一条 GE 链路。显然，在这种情况下，S1 与 S2 之间的 GE 链路是不会发生流量拥塞的。但是，当网络扩建后，S1 接入的用户数增加为 20，如果 S1 与 S2 之间仍然只采用一条 GE 链路，则这条 GE 链路上就可能会出现流量拥塞的情况（因为现在用户带宽的总需求是 2G，但一条 GE 链路只能提供最多 1G 的带宽）。

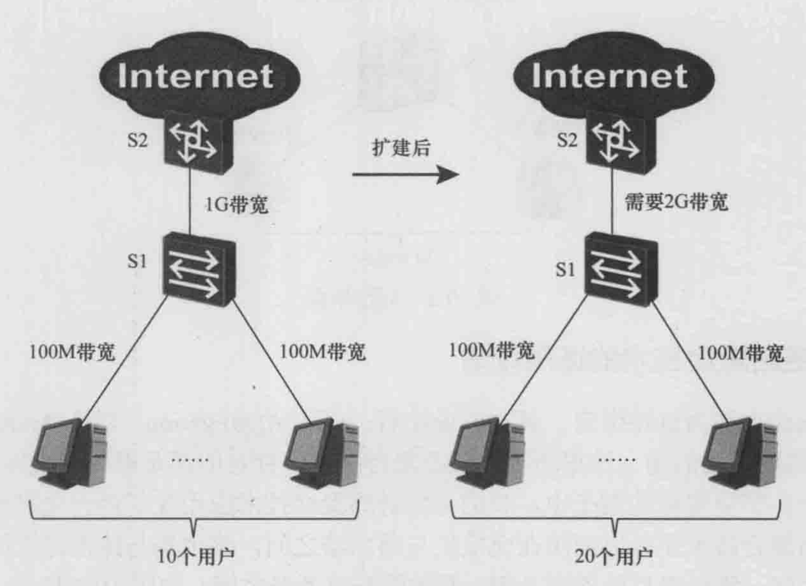


图 10-1 某公司的网络结构

想要解决这个问题，我们可以将 S1 与 S2 之间的链路更换为一条 10GE 链路，但这需要 S1 和 S2 上都有 10GE 端口。如果 S1 或 S2 上没有 10GE 端口，或者根本不支持 10GE 端口，那么就需要更换交换机了。总的来说，这种方法的成本较大，并且 10G 的带宽相对于 2G 的需求来说，实在是富余太多，一定程度上造成了带宽的浪费。还有就是，S1 与 S2 之间如果只有一条链路存在的话，网络的可靠性也会面临很大的威胁。一旦这条链路发生了中断，则所有的用户将完全无法访问 Internet。

针对上面的问题，一个既能满足带宽需求，又能节省成本，而且还能提高 S1 与 S2 连接可靠性的方法便是采用链路聚合技术。例如，如图 10-2 所示，我们可以在 S1 和 S2 之间使用 3 条 GE 链路（当然，S1 和 S2 上都至少需要有 3 个 GE 端口），然后通过链路聚合技术，将这 3 条 GE 链路整合（这里的整合是指逻辑意义上的整合）成为一条最大带宽可达 3G 的逻辑链路（相应地，交换机上的 3 个 GE 端口也被整合成为一个逻辑端口）。一方面，这条逻辑链路可以满足 2G 的带宽需求，另一方面，当某条 GE 链路发生故障而中断之后，这条逻辑链路仍然存在，只是能够提供的带宽值有所下降，但不会导致所有用户完全不能访问 Internet 的糟糕情况。

简而言之，利用链路聚合技术，我们可以：

（1）根据需要灵活地增加网络设备之间的带宽供给；

（2）增强网络设备之间连接的可靠性；

（3）节约成本。

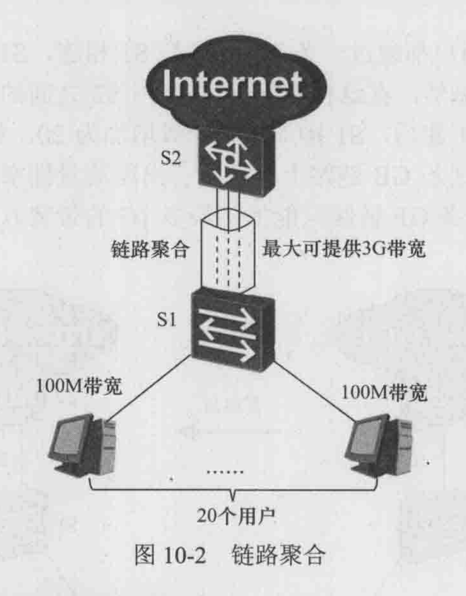


图 10-2 链路聚合

10.1.2 链路聚合技术的适用场景

链路聚合也称为链路绑定，英文的说法有：Link Aggregation、Link Trunking、Link Bonding。需要说明的是，这里所说的链路聚合技术，针对的都是以太网链路。

在上一小节里提到的例子中，我们是将链路聚合技术应用在了两台交换机之间。事实上，链路聚合技术还可以应用在交换机与路由器之间，路由器与路由器之间，交换机与服务器之间，路由器与服务器之间，服务器与服务器之间，如图 10-3 所示。注意，从理论上讲，个人计算机（PC）上也是可以实现链路聚合的，但实际上考虑到成本等因素，没人会在现实中去真正实现。另外，从原理性角度来看，服务器不过就是高性能的计算机。但从网络应用的角度来看，服务器的地位是非常重要的，我们必须保证服务器与其他设备之间的连接具有非常高的可靠性。因此，服务器上经常需要用到链路聚合技术。

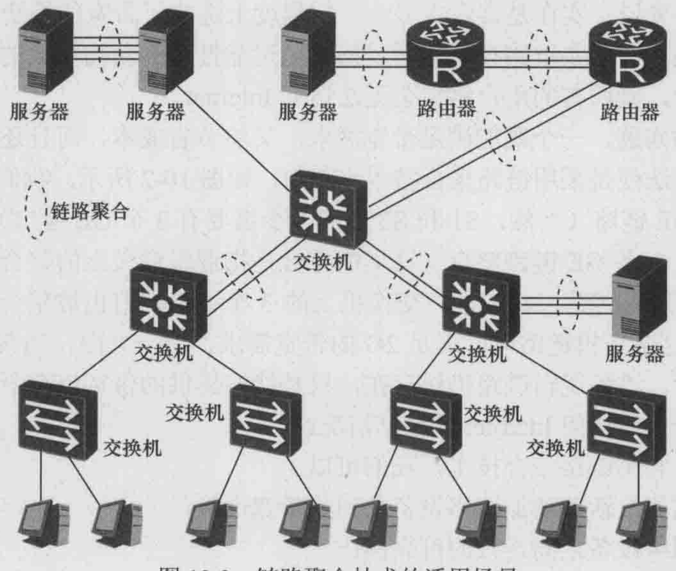


图 10-3 链路聚合技术的适用场景

10.1.3 链路聚合的基本原理

图 10-4 显示的是两台交换机之间的链路聚合情况，我们将以它为例子来说明链路聚合的基本原理。从图 10-4 中我们可以看到，总共有 N 条物理链路被聚合成了一条逻辑链路。通常，我们把聚合后得到的逻辑链路称为聚合链路，而把聚合链路中的每一条物理链路称为成员链路。相应地，我们把聚合后得到的逻辑端口称为聚合端口，而把聚合端口中的每一个物理端口称为成员端口。另外，聚合链路也称为 Eth-Trunk 链路（注：其中的 Eth 是 Ethernet 的简写），聚合端口也称为 Eth-Trunk 端口。

需要说明的是，虽然从理论上讲，同一聚合链路中的各成员链路的带宽可以是不相同的，但在实际中，由于实现难度和实现成本等方面的原因，我们总是要求各成员链路的带宽保持一致。在以下的分析和描述中，我们假定同一聚合链路中各成员链路的带宽总是相同的。

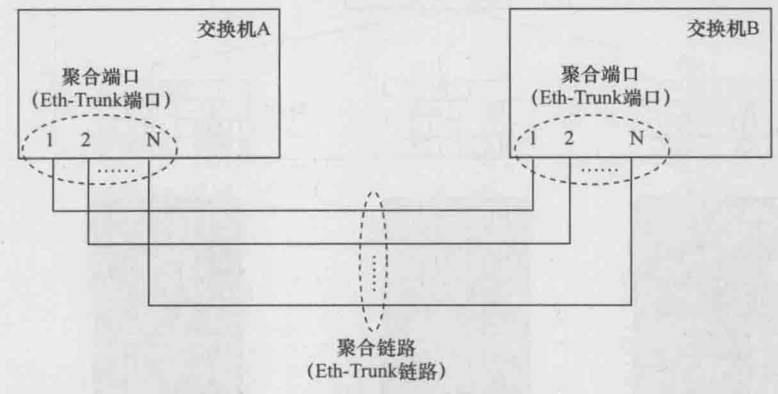


图 10-4 两台交换机之间的链路聚合

现在，我们来看看交换机 A 是如何利用自己的 Eth-Trunk 端口向交换机 B 的 Eth-Trunk 端口发送帧的，如图 10-5 所示。首先，来自交换机 A 的其他端口的帧进入到 Eth-Trunk 端口的帧发送队列。然后，Eth-Trunk 端口的帧分发器（Frame Distributor，FD）将这些帧按照某种算法依次分发给成员端口。FD 的分发顺序为：先将 Frame a 分发给某个成员端口，再将 Frame b 分发给某个成员端口，再将 Frame c 分发给某个成员端口，以此类推。最后，每个成员端口会按照常规方式将来自 FC 的帧发送到自己的物理链路上去。注意，图 10-5 中没有显示出 Eth-Trunk 端口的帧收集器（Frame Collector，FC）和帧接收队列。显然，如果 FD 能够足够均匀地将帧分发给不同的成员端口，那么 Eth-Trunk 端口的带宽就等于各成员端口带宽的总和，相应地，Eth-Trunk 链路的带宽就等于各成员链路带宽的总和。然而，在实际实现中，FD 对帧的分发不可能那么均匀，所以 Eth-Trunk 链路实际能够提供的最大带宽一般会小于各成员链路带宽的总和。

我们再来看看交换机 B 是如何利用自己的 Eth-Trunk 端口接收来自交换机 A 的 Eth-Trunk 端口发送的帧的，如图 10-6 所示。首先，每个成员端口会按照常规方式接收来自物理链路上的帧，接收到的帧都会被送往 Eth-Trunk 端口的帧收集器（Frame Collector，FC）。某一个帧完全进入 FC 后（注：所谓“完全进入”，是指这个帧的末尾都已经进入了 FC），FC 就会把它送往 Eth-Trunk 端口的帧接收队列。图 10-6 显示，最先完全进入 FC 的帧是 Frame a，其次是 Frame b，再其次是 Frame c，如此等等。最后，

Eth-Trunk 端口的帧接收队列中的帧会被依次送往交换机 B 的其他端口。注意，图 10-6 中没有显示出 Eth-Trunk 端口的帧分发器（Frame Distributor，FD）和帧发送队列。

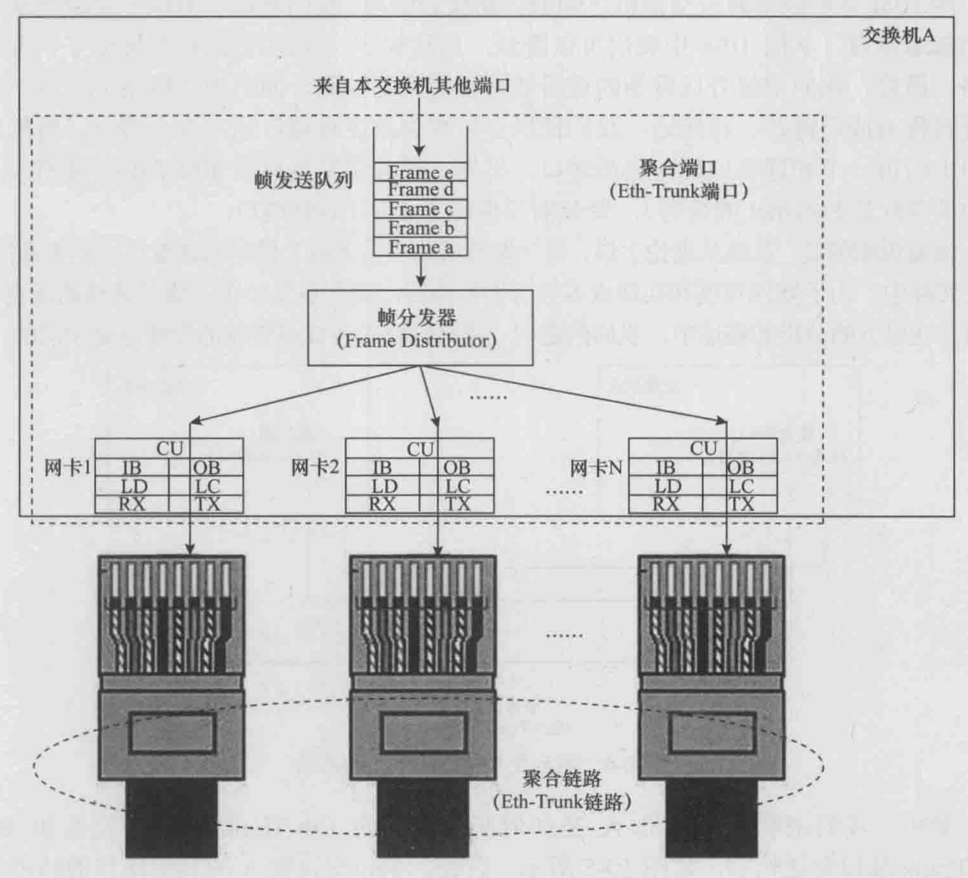


图 10-5　交换机 A 通过聚合端口发送帧

从上面的描述中我们可以看到，链路聚合的基本原理其实就是“流量分担”原理：多条成员链路共同分担了聚合链路的总流量。另外，如果聚合链路中的某条成员链路发生了故障而中断，则聚合链路的总流量会继续被其他成员链路来分担（或者说，本该由故障链路分担的流量将会被 FD 转移给其他的成员链路）。

链路聚合技术看似非常简单，其实并非如此。链路聚合技术需要面临的一个主要问题是“乱序”问题。我们先来说明一下什么是乱序问题。

如图 10-7 所示，交换机 A 的帧发送队列中，帧的先后排列顺序是：a、b、c、d、e。假设 FD 将 Frame a 分发给了成员链路 1，将 Frame b 分发给了成员链路 2，将 Frame c 分发给了成员链路 3，将 Frame d 分发给了成员链路 1，将 Frame e 分发给了成员链路 1。再假设 Frame a 是一个长度较长的帧，而 Frame b 和 Frame c 都是长度较短的帧。由于 Frame b 和 Frame c 的长度较短，所以它们需要的传输时间也就较短，而 Frame a 需要的传输时间相对较长。这样一来，就完全可能会出现这样的情况：Frame b 最先完全进入交换机 B 的 FC，然后是 Frame c，然后是 Frame a，然后是 Frame d，然后是 Frame e。最后，这些帧在交换机 B 的帧接收队列中的排序就成了：b、c、a、d、e。显然，交换机 B 的帧接收队列中帧

的排序不同于它们在交换机 A 的帧发送队列中的排序，这种现象就称为帧的乱序现象。

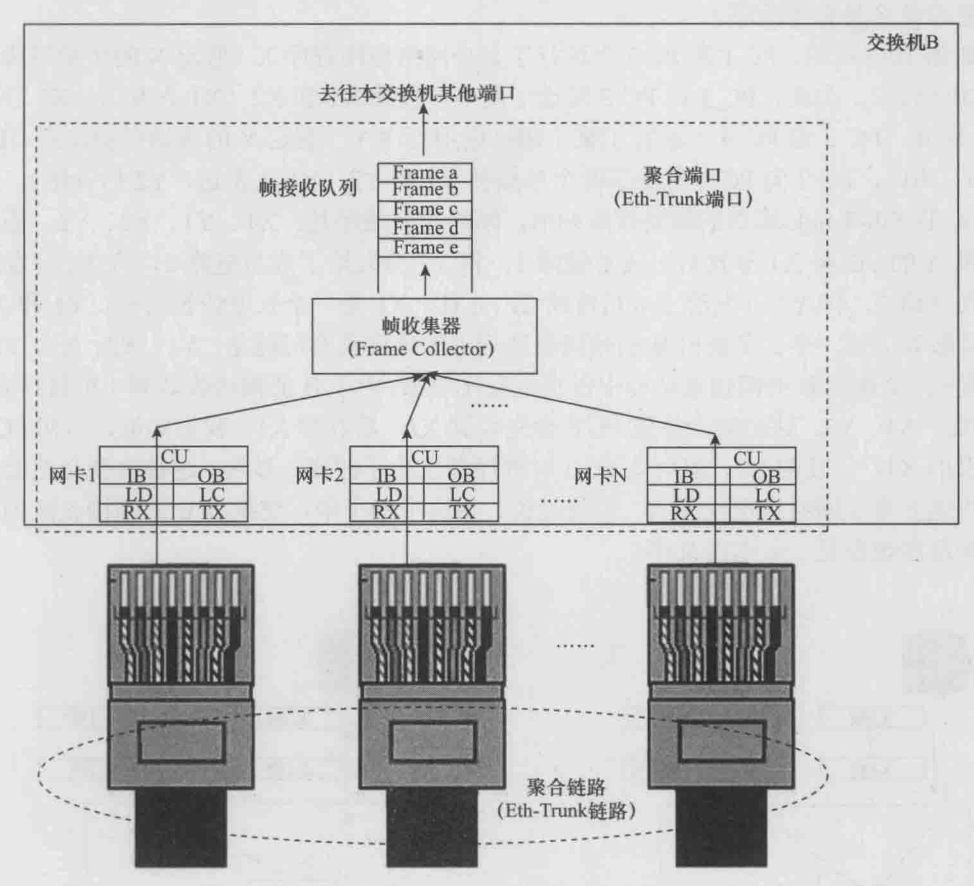


图 10-6　交换机 B 通过聚合端口接收帧

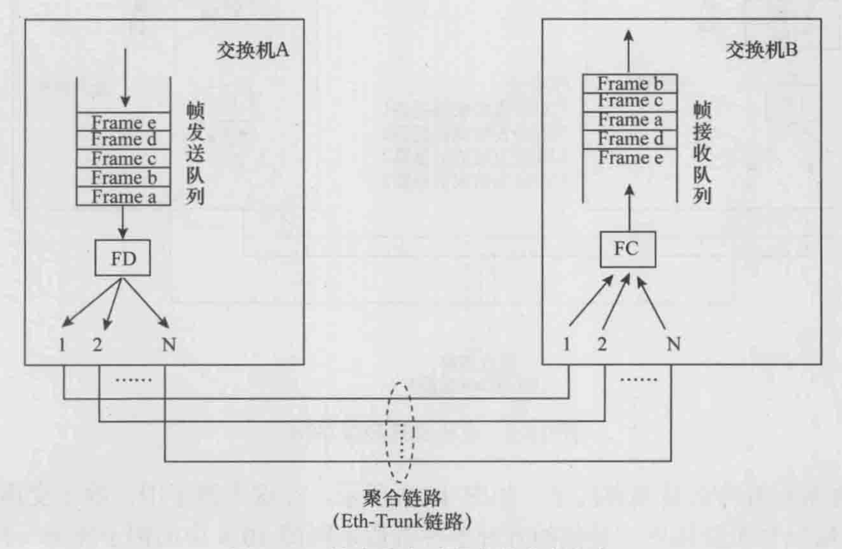


图 10-7　链路聚合过程中的乱序现象

乱序现象又分两种情况，一种是“有害”的乱序，另一种是“无害”的乱序。我们先来看看什么是有害乱序。

如图 10-8 所示，PC 1 和 PC 3 上运行了某个网络应用程序 X（假定 X 的传输层协议是 UDP 协议）。为此，PC 1 向 PC 3 发送了两个单播帧 X1 和 X2（X1 先发送，X2 后发送）。同时，PC 2 和 PC 4 上运行了某个网络应用程序 Y（假定 Y 的传输层协议是 UDP 协议）。为此，PC 2 向 PC 4 发送了两个单播帧 Y1 和 Y2（Y1 先发送，Y2 后发送）。交换机 A 的 Eth-Trunk 端口的帧发送队列中，帧的先后顺序是：X1、Y1、X2、Y2。假设交换机 A 的 FD 将 X1 分发给了成员链路 1，将 Y1 分发给了成员链路 2，将 X2 分发给了成员链路 2，将 Y2 分发给了成员链路 2，并且，X1 是一个长度较长的帧，Y1 和 X2 都是比较短的帧，那么交换机 B 的帧接收队列中的排序就有可能是：Y1、X2、X1、Y2。也就是说，交换机 B 的帧接收队列中发生了乱序现象。由于 B 的帧接收队列中的排序是：Y1、X2、X1、Y2，这必然会导致 PC 3 会先收到 X2，后收到 X1。我们知道，当初 PC 1 是先发的 X1，后发的 X2，但到达 PC 3 时顺序却发生了改变。显然，这种改变必然会或多或少地有害于网络应用程序 X。也就是说，在这个例子中，交换机 B 的帧接收队列中发生的乱序现象是一种有害乱序。

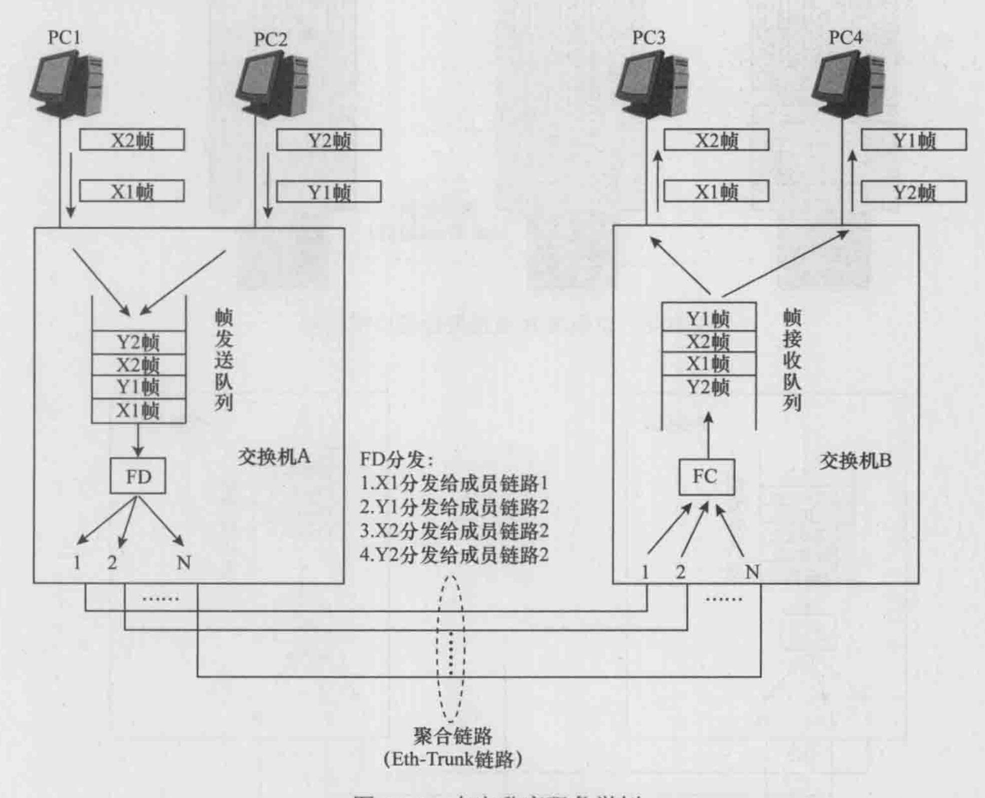


图 10-8 有害乱序现象举例

我们再来看看什么是无害乱序。如图 10-9 所示，在这个例子中，除了交换机 A 的 FD 的分发情况有所变化外，其他的各种条件都假定跟图 10-8 中的例子完全一样。这一次，FD 的分发情况是：将 X1 分发给了成员链路 1，将 Y1 分发给了成员链路 2，将 X2

分发给了成员链路1，将Y2分发给了成员链路2。由于X1是一个较长的帧，所以需要的传输时间较长，所以Y1最先进入了交换机B的FC。接着，Y2也进入了交换机B的FC。然后，X1才进入交换机B的FC，最后是X2（注意，X2不可能比X1先进入交换机B的FC）。虽然，与交换机A的帧发送队列中的顺序相比，交换机B的接收队列中的帧排列顺序已经发生了改变，但是这种改变并不会影响到上层应用。从图10-9中我们可以看到，PC 3先收到X1，后收到X2；PC 4先收到Y1，后收到Y2。也就是说，在这个例子中，交换机B的帧接收队列中发生的乱序现象是一种无害乱序。

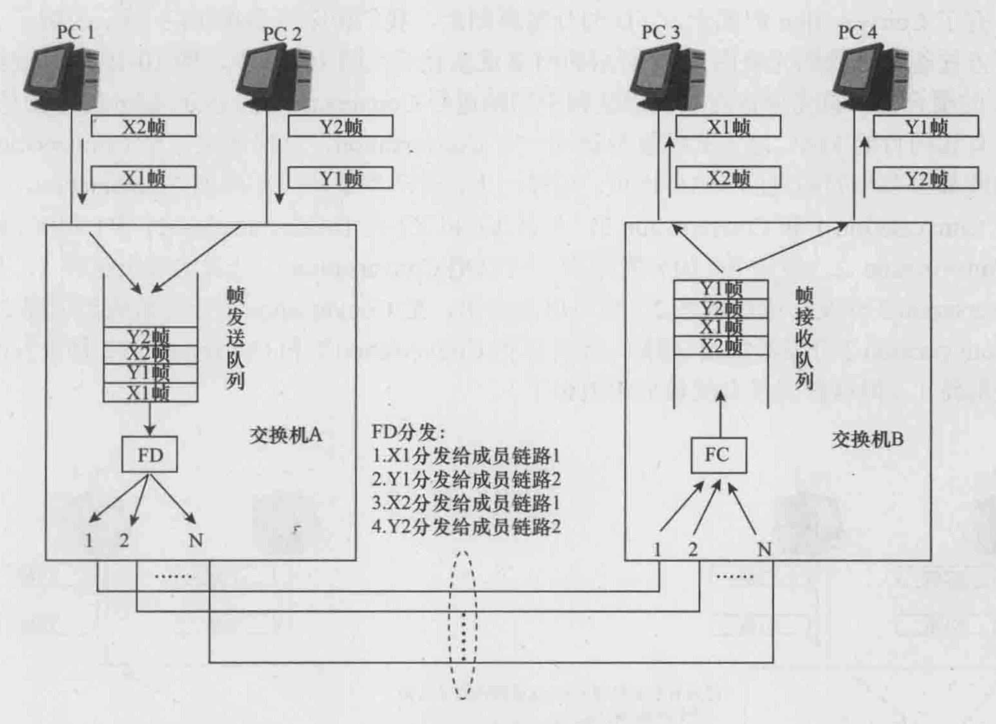


图10-9 无害乱序现象举例

在明白了什么是有害乱序现象和什么是无害乱序现象后，我们来进行一个简要的总结。聚合链路在工作过程中，由于帧的长度有长有短，于是帧的传输时间就有长有短，而不同的帧所经过的成员链路又可能不同，所以一般情况下总是会出现乱序现象。我们无法避免乱序现象，但我们必须避免有害乱序现象。

是否能避免有害乱序现象，关键是看聚合端口的FD是如何将帧分发给不同的成员端口的。为此，人们引入了Conversation这个概念。一个Conversation，是指由若干个帧组成的一个集合，该集合中的不同的帧在接收端的聚合端口的帧接收队列中的先后顺序必须与它们在发送端的聚合端口的帧发送队列中的先后顺序保持一致。如果保持了一致，则一定不会发生有害乱序现象；如果没有保持一致，则一定会发生有害乱序现象。需要强调的是，不同的Conversation之间的交集必须是空集。也就是说，同一个帧，不能既属于这个Conversation，又属于另外一个Conversation。还有就是，一个帧不能不属于任何Conversation。

为了避免有害乱序现象的产生，同时又能实现流量分担，聚合端口的FD必须遵从如下的分发原则。

（1）同一个 Conversation 中的帧，必须被分发给同一条成员链路（这样就避免了有害乱序现象）。

（2）不同 Conversation 中的帧，可以被分发给同一条成员链路，也可以被分发给不同的成员链路（这样就实现了流量分担）。

从上述 FD 的分发原则来看，同一个 Conversation 中的帧是不会乱序的，这就避免了有害乱序现象的产生。另一方面，不同 Conversation 中的帧是有可能会乱序的，但这种乱序只是无害乱序。

有了 Conversation 的概念及 FD 的分发原则后，我们再来看看图 10-9 所示的例子。为了方便起见，我们先将图 10-9 所示的网络重新显示在图 10-10 中。图 10-10 中，交换机 A 的聚合端口首先应该对帧发送队列中的帧进行 Conversation 的划分，划分的方法是：把具有相同目的 MAC 地址的帧划分进同一个 Conversation，且保证同一个 Conversation 中的帧都具有相同的目的 MAC 地址。这样一来，就产生了两个不同的 Conversation，分别为 Conversation 1 和 Conversation 2，并且 X1 和 X2 属于 Conversation 1，Y1 和 Y2 属于 Conversation 2。根据 FD 的分发原则，可以把 Conversation 1 分发给成员链路 1，把 Conversation 2 分发给成员链路 2；也可以反过来，把 Conversation 1 分发给成员链路 2，把 Conversation 2 分发给成员链路 1；还可以把 Conversation 1 和 Conversation 2 都分发给成员链路 1（但这样就没有流量分担效果了）。

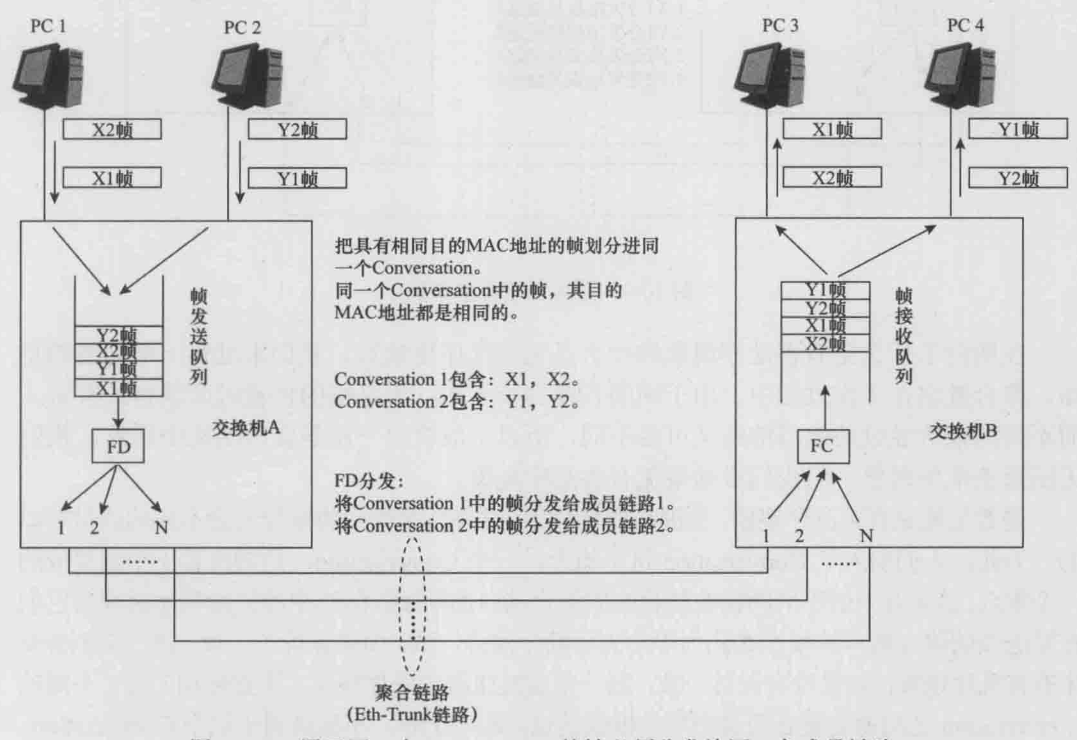


图 10-10 属于同一个 Conversation 的帧必须分发给同一条成员链路

关于聚合端口的 FD 的分发原则，图 10-11 显示了一种更为普遍的情况。在图 10-11 中，如果成员链路 4 发生了中断，则 FD 会将 Conversation 5 分发给成员端口 2，将

Conversation 6 分发给成员端口 3。

在实际实现链路聚合时，聚合链路的 FD 需要根据一种 HASH 算法来定义出恰当的 Conversation，然后再对不同的 Conversation 进行分发。定义出恰当的 Conversation 并不是一件容易的事情。在图 10-10 所示的例子中，帧的目的 MAC 地址被选择成为了用来定义 Conversation 的参考量。然而，在实际的网络环境中，聚合链路两端的设备属性（例如，交换机跟交换机聚合，路由器跟路由器聚合，服务器跟服务器聚合，交换机跟路由器聚合，交换机跟服务器聚合，如此等等，见图 10-3）以及上层应用的属性，都需要成为确定 Conversation 的参考量的考虑因素，而最终的参考量可能是目的 MAC 地址，也可能是源 MAC 地址，也可能是目的 IP 地址，也可能是源 IP 地址，也可能是几种不同地址的组合，还可能是上层协议中的某些参数，如此等等。

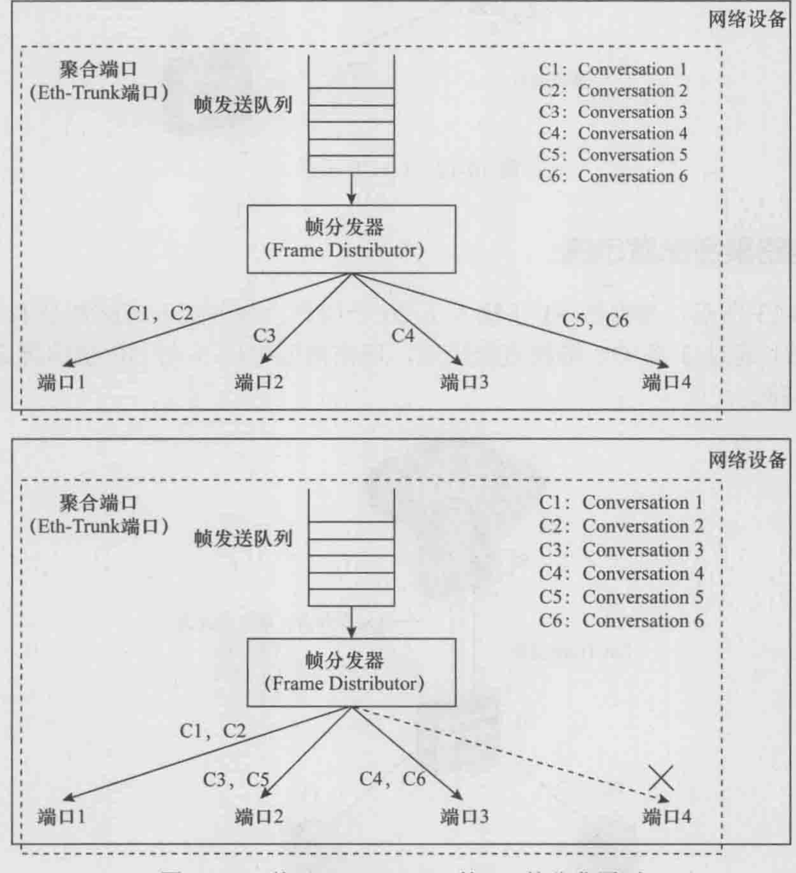


图 10-11　基于 Conversation 的 FD 的分发原则

10.1.4　LACP

LACP 即链路聚合控制协议，它是 Link Aggregation Control Protocol 的简称。该协议定义在 IEEE 802.3ad 中（IEEE 802.3ad 包含了 LACP 和 Marker Protocol 这两个协议）。我们这里省去对 LACP 协议的具体描述，但读者应该知道 LACP 是一个关于链路聚合技术的协议规范。

在设备上实现链路聚合时，通常可以有两种模式：一种称为手工负载分担模式；另一种称为 LACP 模式。显然，LACP 模式实现起来会增加设备本身的复杂度，但它的自动化程度更高，并且可以避免一些人为的错误。例如，在图 10-12 中，如果采用手工模式在交换机 S1 和 S2 上配置聚合端口，就有可能在 S1 上绑定了 4 个端口，而在 S2 上绑定了 3 个端口，这种错误有时并不是那么容易被发现的。然而，如果采用 LACP 模式，则 S1 和 S2 之间会通过交换 LACP 协议帧的方式进行自动协商，从而很容易发现问题所在。

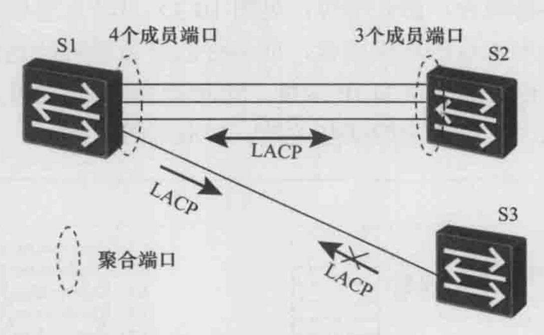


图 10-12 LACP 示意

10.1.5 链路聚合配置示例

如图 10-13 所示，交换机 S1 下接入了 20 个用户，每条接入链路都是 FE 链路。交换机 S2 与 S1 通过 3 条 GE 链路直接相连，现在需要将这 3 条 GE 链路绑定成为一条 Eth-Trunk 链路。

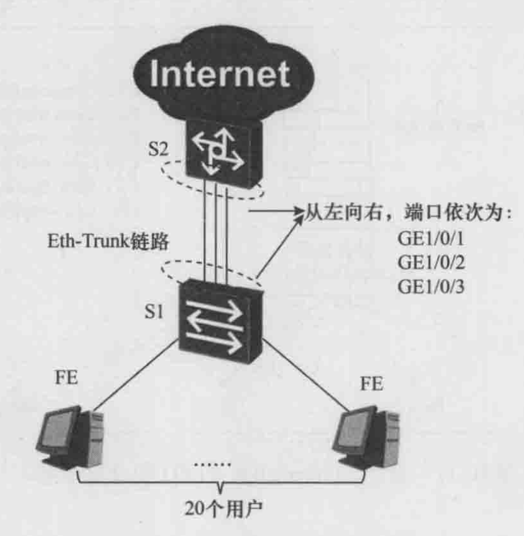


图 10-13 Eth-Trunk 配置示例

1. 配置思路

（1）创建 Eth-Trunk 端口。

（2）（可选）配置链路聚合模式。

（3）将物理端口加入进 Eth-Trunk 端口。

（4）配置二层链路的连通性（如 VLAN 配置等等）。

2. 配置步骤

#在 S1 上创建编号为 1 的 Eth-Trunk 端口（Eth-Trunk1）。

```
<Quidway> system-view
[Quidway] sysname S1
[S1] interface Eth-Trunk1
[S1-Eth-Trunk1]
```

#在 S2 上创建编号为 1 的 Eth-Trunk 端口（Eth-Trunk1）。注意，Eth-Trunk 端口的编号在两端的设备上需保持一致。

```
<Quidway> system-view
[Quidway] sysname S2
[S2] interface Eth-Trunk1
[S2-Eth-Trunk1]
```

#（可选）在 S1 上配置 Eth-Trunk1 端口的工作模式为手工负载分担模式。

```
[S1-Eth-Trunk1] mode manual load-balance
```

#（可选）在 S2 上配置 Eth-Trunk1 端口的工作模式为手工负载分担模式。

```
[S2-Eth-Trunk1] mode manual load-balance
```

Eth-Trunk 端口的工作模式分为手工负载分担模式和 LACP 模式两种，可以使用命令 **mode**{**lacp**|**manual load-balance**}来进行配置。缺省情况下，Eth-Trunk 端口的工作模式为手工负载分担模式。配置时需要注意，Eth-Trunk 端口的工作模式在两端的设备上必须保持一致。在将任何成员端口加入进 Eth-Trunk 端口之前，必须先配置好 Eth-Trunk 端口的工作模式。

#在 S1 上，将物理端口 GE1/0/1、GE1/0/2、GE1/0/3 加入进 Eth-Trunk1 端口。

```
[S1-Eth-Trunk1] trunkport gigabitethernet 1/0/1 to 1/0/3
[S1-Eth-Trunk1] quit
```

#在 S2 上，将物理端口 GE1/0/1、GE1/0/2、GE1/0/3 加入进 Eth-Trunk1 端口。

```
[S2-Eth-Trunk1] trunkport gigabitethernet 1/0/1 to 1/0/3
[S2-Eth-Trunk1] quit
```

将物理端口加入进 Eth-Trunk 时还需要注意，加入同一个 Eth-Trunk 端口的物理端口必须是同一类型的端口，并且其属性需要保持完全一致（例如，这些端口都属于同一个 VLAN）。

#配置 S1 的 Eth-Trunk1 端口，允许属于 VLAN 1000 的帧通过。

```
[S1] interface Eth-Trunk1
[S1-Eth-Trunk1] port link-type trunk
[S1-Eth-Trunk1] port trunk allow-pass vlan 1000
```

#配置 S2 的 Eth-Trunk1 端口，允许属于 VLAN 1000 的帧通过。

```
[S2] interface Eth-Trunk1
[S2-Eth-Trunk1] port link-type trunk
[S2-Eth-Trunk1] port trunk allow-pass vlan 1000
```

我们可以使用 **display eth-trunk** [*trunk-id* [**interface** *interface-type interface-number* | **verbose**]]命令来查看 Eth-Trunk 端口的配置信息，从而可以对所做的配置进行验证。以 S1 为例。

#在 S1 上查看 Eth-Trunk1 端口的配置信息。

```
[S1] display eth-trunk 1 verbose
Eth-Trunk1's state information is :
```

```
WorkingMode:  NORMAL          Hash arithmetic:  According to SIP-XOR-DIP
Least Active-linknumber : 1   Max Bandwidth-affected-linknumber : 8
Operate status : up           Number Of Up Port In Trunk : 0
--------------------------------------------------------------------------------
PortName                      Status      Weight
GigabitEthernet1/0/1          Up          1
GigabitEthernet1/0/2          Up          1
GigabitEthernet1/0/3          Up          1

Flow statistic
Interface GigabitEthernet1/0/1
    Last 300 seconds input rate 32 bits/sec, 0 packets/sec
    Last 300 seconds output rate 32 bits/sec, 0 packets/sec
    148 packets input, 18944 bytes, 0 drops
    246 packets output, 31488 bytes, 0 drops
 Interface GigabitEthernet1/0/2
    Last 300 seconds input rate 32 bits/sec, 0 packets/sec
    Last 300 seconds output rate 32 bits/sec, 0 packets/sec
    147 packets input, 18816 bytes, 0 drops
    246 packets output, 31488 bytes, 0 drops
 Interface GigabitEthernet1/0/3
    Last 300 seconds input rate 56 bits/sec, 0 packets/sec
    Last 300 seconds output rate 48 bits/sec, 0 packets/sec
    144 packets input, 18432 bytes, 0 drops
    174 packets output, 22272 bytes, 0 drops
 Interface Eth-Trunk1
    Last 300 seconds input rate 96 bits/sec, 0 packets/sec
    Last 300 seconds output rate 96 bits/sec, 0 packets/sec
    439 packets input, 56192 bytes, 0 drops
    666 packets output, 85248 bytes, 0 drops
```

在上面的回显信息中，“WorkingMode： NORMAL”表示 Eth-Trunk1 端口的工作模式为 NORMAL，即手工负载分担模式（如果显示 LACP，则表示工作模式为 LACP 模式）。“Least Active-linknumber：1”表示处于 Up 状态的成员链路的下限阈值为 1。“Operate status：up”表示 Eth-Trunk1 端口的状态为 Up。从 Flow statistic 下面的信息可以看出，Eth-Trunk1 端口包含了 3 个成员端口，分别是 GigabitEthernet1/0/1、GigabitEthernet1/0/2、GigabitEthernet1/0/3，其中每个端口均转发了一定的流量，而 Eth-Trunk1 端口总的转发量正是各个成员端口的转发量的总和。

10.2 Smart Link

10.2.1 Smart Link 的基本原理

如图 10-14 所示，接入交换机 S4 下面接入了 *N* 个用户终端，S4 通过两条上行链路 Link2-4 和 Link3-4 分别与汇聚交换机 S2 和 S3 相连。S2 和 S3 分别通过链路 Link1-2 和 Link1-3 与核心交换机 S1 相连，S1 通过路由器接入 Internet。为了消除工作环路，每台交换机上都运行了 STP 协议。假设 STP 树的链路包含了 Link1-2、Link1-3、Link2-4，那么，当 Link2-4 中断后，Link3-4 就会加入到 STP 树中，从而保证了网络

的连通性。

然而我们知道，STP 的收敛速度是比较慢的，一般在秒的数量级上。如果网络中的链路是一些高速链路，那么在 STP 切换链路的过程中，就会导致大量的数据丢失。如果用户终端上运行了一些对丢包非常敏感的业务，那么这些业务就会受到严重的影响。

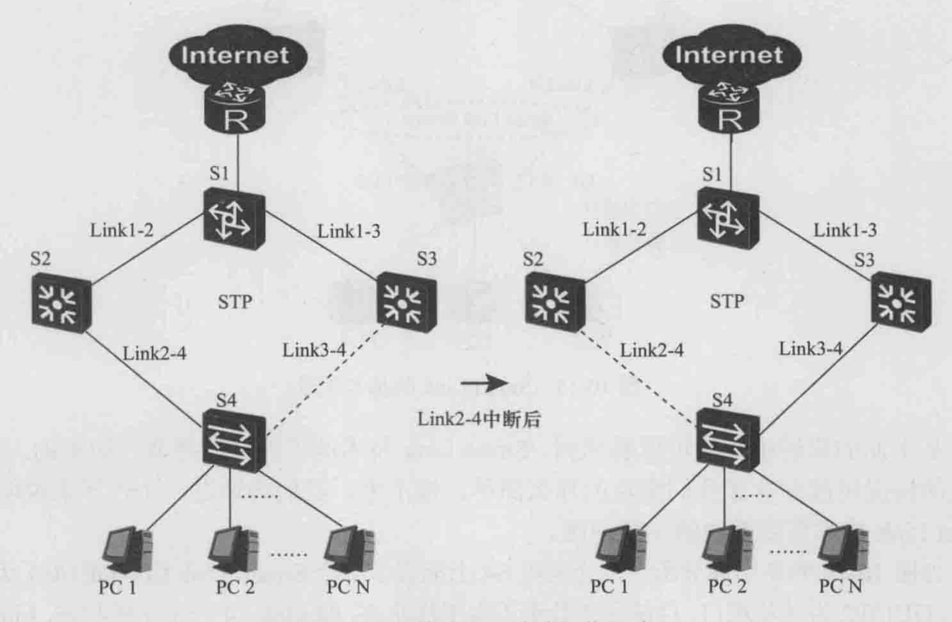


图 10-14　利用 STP 消除工作环路

针对上述问题，华为公司设计并实现了一种被称为 Smart Link 的私有协议，该协议的主要作用是在一定的场景下替代 STP 协议，并能实现快速（毫秒级）的链路切换。

一个 Smart Link 组由两个端口组成，其中一个为主端口，另一个为从端口。正常情况下，只有主端口处于转发（Active）状态，而从端口被阻塞，处于待命（Inactive）状态。当主端口发生故障时，Smart Link 组会自动将主端口阻塞，并立即将从端口的状态从待命状态切换到转发状态。Smart Link 技术常用于双上行组网环境。

如图 10-15 所示，交换机 S4 上配置了一个 Smart Link 组，GE1/0/1 为其主端口，GE1/0/2 为其从端口。正常情况下，主端口 GE1/0/1 处于转发状态，从端口 GE1/0/2 处于待命状态，所以真正处于工作状态的链路有 Link1-3、Link1-2、Link2-4，而 Link3-4 则处于中断状态，这就避免了环路的产生。如果主端口 GE1/0/1 本身突然发生故障，或者主端口 GE1/0/1 感知到了 Link2-4 的中断，那么 Smart Link 组就会立即将主端口 GE1/0/1 设定为阻塞状态，同时立即将从端口 GE1/0/2 的状态从待命状态切换到转发状态。这样一来，真正处于工作状态的链路就立即变成了 Link1-3、Link1-2、Link3-4，而 Link2-4 则处于中断状态。这样一来，既保证了网络的连通性，又避免了任何环路的产生。注意，Smart Link 协议是与 STP 协议互斥的，所以图 10-15 所示的网络中是没有运行 STP 的。

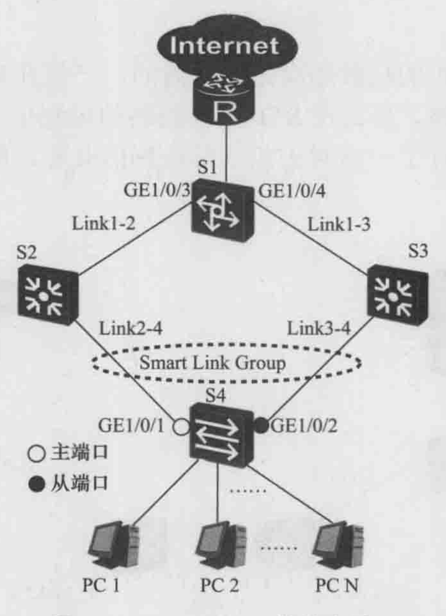


图 10-15 Smart Link 的基本原理

从上面的描述中我们可以感觉到，Smart Link 技术的工作原理是非常简单的。然而，真正的情况可能并非像我们想象的那么简单。接下来，我们将通过一个例子来说明一下 Smart Link 技术需要解决的主要问题。

如图 10-16 的左半部分所示，交换机 S4 上配置了一个 Smart Link 组，GE1/0/1 为其主端口，GE1/0/2 为其从端口，目前网络处于正常工作状态，即 Link3-4 处于中断状态，Link1-3、Link1-2、Link2-4 都处于工作状态。另外，我们假定 PC 1 的网口的 MAC 地址为 MAC-1。

假设在 *T* 时刻，PC 1 向 Internet 发送了一个帧，那么这个帧必然会经过 Link2-4 和 Link1-2，然后从交换机 S1 的 GE1/0/3 端口进入 S1，S1 会将这个帧转发给路由器。根据交换机的 MAC 地址学习机制，几乎也是在 T 时刻（忽略掉这个帧从 PC 1 运动到 S1 所经历的时间），S1 上的关于 MAC-1 的表项的内容将成为：对应的端口为 GE1/0/3，老化计时器（倒数计时器）的值为 300 秒（缺省值）。

然后，如图 10-16 的右半部分所示，我们假设在 *T*+5 秒的时刻，Link2-4 发生了中断，S4 的主端口 GE1/0/1 立即被阻塞，从端口 GE1/0/2 立即被切换成转发状态。这时的工作链路变成了 Link1-3，Link1-2，Link3-4。同时，S1 上的关于 MAC-1 的表项的内容将成为：对应的端口为 GE1/0/3，老化计时器的值为 295 秒。

现在，我们假设时间已经从 *T*+5 秒时刻过渡到了 *T*+10 秒时刻，并且假设在这段时间内 PC 1 没有向外发送过任何帧，因此，S1 上的 MAC 地址表中仍然存在关于 MAC-1 的表项，MAC-1 对应的端口仍然为 GE1/0/3，只是老化计时器的值已经变成了 290 秒，如图 10-17 所示。就在 *T*+10 秒这个时刻，我们假设 S1 从路由器那里接收到了一个目的 MAC 地址为 MAC-1 的帧。显然，S1 在查询了自己的 MAC 地址表后，会将这个帧从其 GE1/0/3 端口转发出去，而不是从其 GE1/0/4 端口转发出去。然而我们知道，此时 Link2-4 是处于中断状态的，所以这个帧是不可能被送达至 PC 1 的，这就发生了我们不愿看到的丢帧现象。一个极端的情况是，假设在 *T*+10 秒时刻至 *T*+300 秒时刻这段时间内，PC 1 一直都没

有向外发送过帧，也就是说，S1 上的 MAC 地址表中 MAC-1 对应的端口一直是 GE1/0/3，那么在这段时间内，路由器向 S1 发送的、目的 MAC 地址为 MAC-1 的所有帧都会丢失。

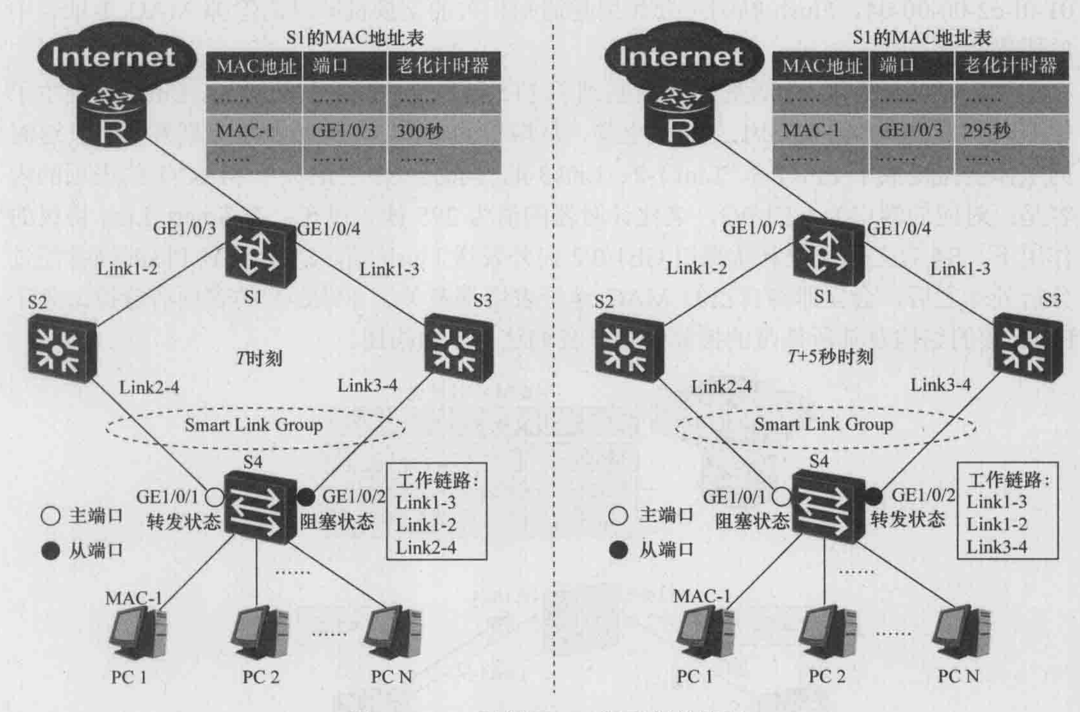


图 10-16　*T* 时刻和 *T*+5 秒时刻的情况

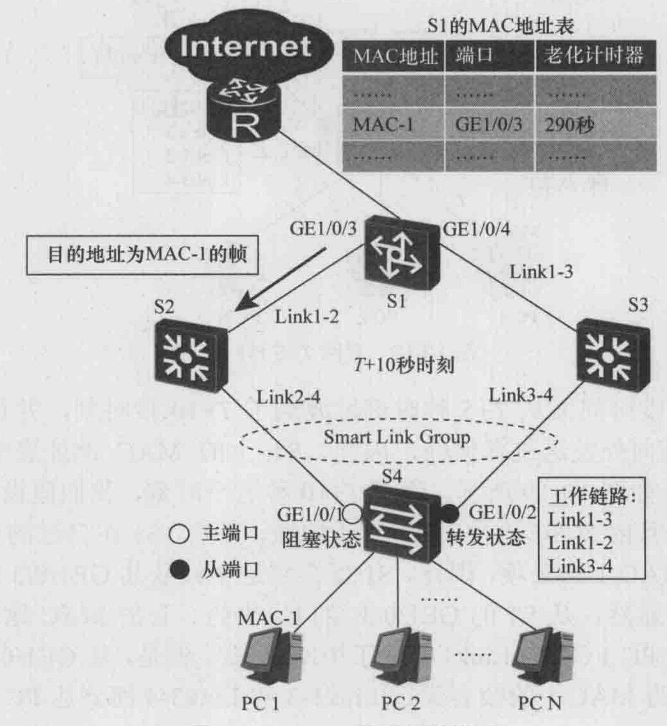


图 10-17　*T*+10 秒时刻的情况

Smart Link 是如何避免上述丢帧现象的呢？针对上述丢帧问题，Smart Link 定义了一种被称为 Flush 帧的协议帧，这种帧的目的 MAC 地址为组播 MAC 地址 01-0f-e2-00-00-04。Flush 帧的主要作用是通知相关的交换机即时清除掉 MAC 地址表中的错误表项。

如图 10-18 所示，假设时间重新回到了 *T*+5 秒那一时刻。在此时刻，Link2-4 发生了中断，S4 的主端口 GE1/0/1 立即被阻塞，从端口 GE1/0/2 立即被切换成转发状态。这时的工作链路变成了 Link1-3、Link1-2、Link3-4。同时，S1 上的关于 MAC-1 的表项的内容是：对应的端口为 GE1/0/3，老化计时器的值为 295 秒。现在，在 Smart Link 协议的作用下，S4 会立即通过其从端口 GE1/0/2 向外发送 Flush 帧，S1 接收到 Flush 帧并经过分析处理之后，会立即将自己的 MAC 地址表中那条关于 MAC-1 的表项清除掉。关于 Flush 帧的结构及其所携带的控制信息，我们这里不做描述。

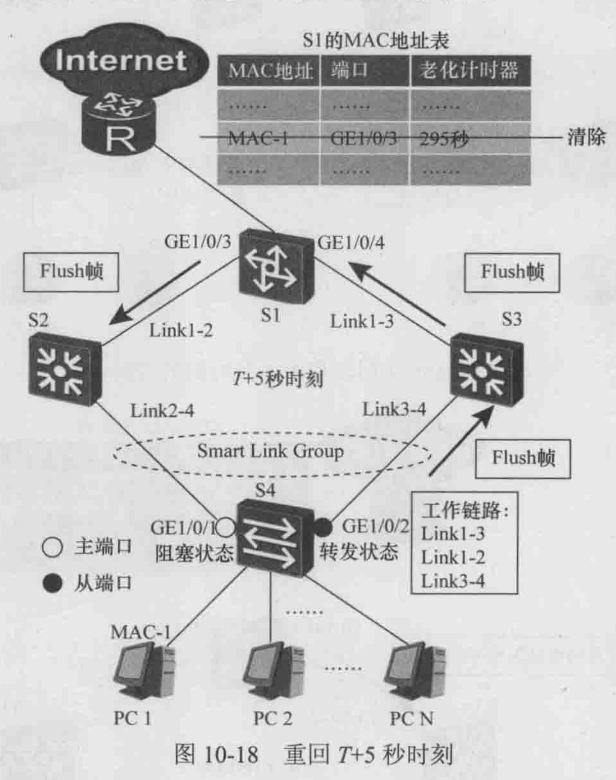


图 10-18　重回 *T*+5 秒时刻

接下来，假设时间又从 *T*+5 秒时刻过渡到了 *T*+10 秒时刻，并且假设在这段时间内 PC 1 没有向外发送过任何帧，因此，S1 上的 MAC 地址表中不会存在关于 MAC-1 的表项，如图 10-19 所示。就在 *T*+10 秒这个时刻，我们假设 S1 从路由器那里接收到了一个目的 MAC 地址为 MAC-1 的帧。显然，S1 在自己的 MAC 地址表中查找不到关于 MAC-1 的表项，因此，S1 就会将这个帧从其 GE1/0/3 端口和 GE1/0/4 端口泛洪出去。显然，从 S1 的 GE1/0/3 端口出去的、目的 MAC 地址为 MAC-1 的帧无法被送达至 PC 1（因为 Link2-4 处于中断状态），但是，从 GE1/0/4 端口出去的、目的 MAC 地址为 MAC-1 的帧会经过 Link1-3 和 Link3-4 而到达 PC 1，这样就避免了丢帧的情况。

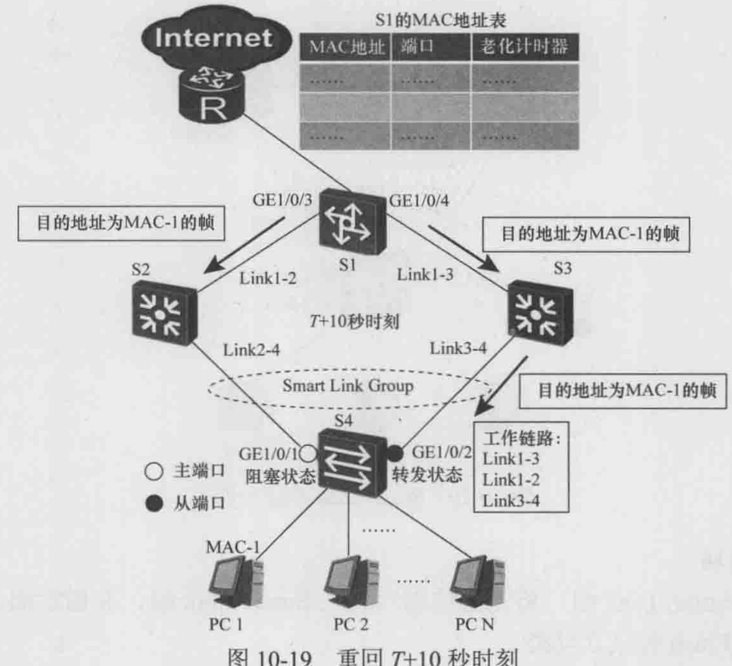


图 10-19　重回 *T*+10 秒时刻

从前面的例子中我们可以看到，Flush 帧在 Smart Link 协议中扮演着非常关键的作用。为了控制 Flush 帧的传播及作用范围，Smart Link 会专门为 Flush 帧定义一个 VLAN，称为控制 VLAN。Flush 帧在被发送之前必须带上控制 VLAN 的 Tag。如果某台设备需要接收并处理 Flush 帧，那么我们就必须事先对该设备进行相应的配置，使它能够接收、识别并处理带有控制 VLAN Tag 的帧。如果一台设备没有进行上述配置，那么它在接收到带有控制 VLAN Tag 的帧时，会直接将其丢弃。

最后，我们简单介绍一下 Smart Link 的回切功能。正常情况下，Smart Link 的主端口处于 Active 状态，从端口处于 Inactive 状态。当主端口 Down 掉（主链路中断）后，主端口的状态会切换成 Inactive，从端口的状态会切换为 Active。但是，主端口重新 Up（主链路重新接通）之后，Smart Link 并不会自动将主端口的状态回切到 Active，同时也不会将从端口的状态回切到 Inactive。如果需要将主端口的状态回切到 Active，将从端口的状态回切到 Inactive，那么我们就必须事先配置好 Smart Link 的回切功能。另外，在配置 Smart Link 回切功能时，我们还需要配置一个被称为“回切时间”的参数，其缺省值为 60 秒。也就是说，主端口虽然重新 Up（主链路重新接通）了，但 Smart Link 还应该等待一段时间（这段时间就是所谓的回切时间）之后才进行回切操作。因为主端口虽然重新 Up（主链路重新接通）了，但其工作状态可能还并不稳定，甚至可能出现闪通和闪断的现象，这就是为什么回切操作一般不宜马上进行的原因。

10.2.2　Smart Link 配置示例

如图 10-20 所示，Switch A、Switch B、Switch C 组成了一个环路。我们需要在 Switch A 上将端口 GE1/0/1 和 GE1/0/2 配置在一个 Smart Link 组内，并让 GE1/0/1 成为主端口，GE1/0/2 成为从端口。

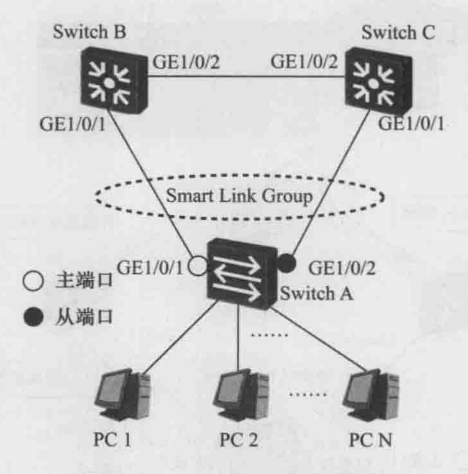


图 10-20　Smart Link 配置示例

1. 配置思路

（1）创建 Smart Link 组，将相应的端口加入 Smart Link 组，并指定端口角色。

（2）使能 Flush 帧发送功能。

（3）使能 Flush 帧接收功能。

（4）使能 Smart Link 回切功能。

（5）使能 Smart Link 功能。

2. 配置步骤

由于 Smart Link 协议是与 STP 协议互斥的，所以在配置 Smart Link 之前需要先进入相应的接口视图，并使用 **stp disable** 命令来取消 STP 功能。

配置 Switch A。

```
[SwitchA] interface gigabitethernet 1/0/1
[SwitchA-GigabitEthernet1/0/1] stp disable
[SwitchA-GigabitEthernet1/0/1] quit
[SwitchA] interface gigabitethernet 1/0/2
[SwitchA-GigabitEthernet1/0/2] stp disable
[SwitchA-GigabitEthernet1/0/2] quit
```

接下来在 Switch A 上创建 Smart Link 组 1，并使用 **port** 命令将 GE1/0/1 配置为 Smart Link 组 1 的主端口，将 GE1/0/2 配置为 Smart Link 组 1 的从端口。

配置 Switch A。

```
[SwitchA] smart-link group 1
[SwitchA-smlk-group1] port gigabitethernet 1/0/1 master
[SwitchA-smlk-group1] port gigabitethernet 1/0/2 slave
```

然后，使用 **flush send** 命令使能 Smart Link 组 1 发送 Flush 帧的功能，携带的控制 VLAN 编号是 10，密码是“123”。

配置 Switch A。

```
[SwitchA-smlk-group1] flush send control-vlan 10 password simple 123
```

在 Switch B 和 Switch C 上使用 **smart-link flush receive** 命令，指定它们的 GE1/0/1 端口和 GE1/0/2 端口可以接收和处理携带控制 VLAN 编号是 10 的 Flush 帧。

配置 Switch B。

```
[SwitchB] interface gigabitethernet 1/0/1
[SwitchB-GigabitEthernet1/0/1] smart-link flush receive control-vlan 10 password simple 123
[SwitchB-GigabitEthernet1/0/1] quit
[SwitchB] interface gigabitethernet 1/0/2
[SwitchB-GigabitEthernet1/0/2] smart-link flush receive control-vlan 10 password simple 123
[SwitchB-GigabitEthernet1/0/2] quit
```

配置 Switch C。

```
[SwitchC] interface gigabitethernet 1/0/1
[SwitchC-GigabitEthernet1/0/1] smart-link flush receive control-vlan 10 password simple 123
[SwitchC-GigabitEthernet1/0/1] quit
[SwitchC] interface gigabitethernet 1/0/2
[SwitchC-GigabitEthernet1/0/2] smart-link flush receive control-vlan 10 password simple 123
[SwitchC-GigabitEthernet1/0/2] quit
```

接下来，使用 **restore enable** 命令配置回切功能，使用 **timer wtr** 命令设定回切时间为 30 秒。

配置 SwitchA。

```
[SwitchA-smlk-group1] restore enable
[SwitchA-smlk-group1] timer wtr 30
```

最后，使用命令 **smart-link enable** 来使能 Smart Link 组 1 的功能。

配置 SwitchA。

```
[SwitchA-smlk-group1] smart-link enable
```

现在，我们需要对配置好的 Smart Link 组 1 进行确认，也就是通过 **display smart-link group** 命令来查看相关信息。以 Switch A 为例。

```
<SwitchA> display smart-link group 1
Smart Link group 1 information ：
  Smart Link group was enabled
  Wtr-time is： 30 sec.
  There is no Load-Balance
  There is no protected-vlan reference-instance
  DeviceID： 0018-2000-0083   Control-vlan ID： 10
  Member                  Role       State ...
  ---------------------------------------------
  GigabitEthernet1/0/1    Master     Active ...
  GigabitEthernet1/0/2    Slave      Inactive...
```

从回显信息中我们可以看到，Smart Link 组 1 已经使能，GigabitEthernet1/0/1 作为主端口处于 Active 状态，GigabitEthernet1/0/2 作为从端口处于 Inactive 状态，控制 VLAN 的 ID 是 10，回切时间是 30 秒。

10.3 Monitor Link

10.3.1 Monitor Link 的基本原理

如图 10-21 所示，交换机 S4 上配置了一个 Smart Link 组，其中 GE1/0/1 为主端口，GE1/0/2 为从端口，GE1/0/1 的状态为 Active，GE1/0/2 的状态为 Inactive。如果此时 S2 的 GE1/0/1 端口发生了故障，导致 Link1-2 中断，那么会出现什么样的后果呢？显然，S4 不可能感知到 S2 的 GE1/0/1 端口发生了故障，于是，从 S4 的主端口 GE1/0/1 发出的

帧都会因此而丢失。

针对上述问题，华为公司设计并实现了一种被称为 Monitor Link 的私有协议，该协议的主要作用是在一定的场景下配合 Smart Link 的使用，从而更好地避免丢帧情况的发生。

图 10-21 中，我们可以在 S2 上配置一个 Monitor Link 组，这个 Monitor Link 组包含了两个端口，一个是 GE1/0/1 端口，其角色是上行端口，另一个是 GE1/0/2 端口，其角色是下行端口。Monitor Link 的工作原理是：一个 Monitor Link 组由一个上行端口和若干个下行端口组成，如果上行端口因种种原因而不能正常工作时，则其所有的下行端口都必须立即被 Down 掉。也就是说，下行端口与上行端口存在一种联动机制，下行端口的工作状态应该与上行端口的工作状态保持一致。

回到图 10-21 中，在正常情况下，处于工作状态的链路有 Link1-3、Link1-2、Link2-4。如果 S2 的 GE1/0/1 端口发生了故障，则在 Monitor Link 协议的作用下，S2 的 GE1/0/2 端口就会立即被 Down 掉，这样一来，S4 的 GE1/0/1 端口也就无法正常工作。于是，S4 的 Smart Link 就会立即进行切换操作，将其从端口 GE1/0/2 的状态从 Inactive 切换到 Active。于是，处于工作状态的链路就变成了 Link1-3 和 Link3-4，网络的连通性仍然得到了保障。

同理，为了进一步增强网络的可靠性，我们还可以在 S3 上也配置一个 Monitor Link 组，使得 S3 的 GE1/0/2 端口可以与 GE1/0/1 端口实现联动。

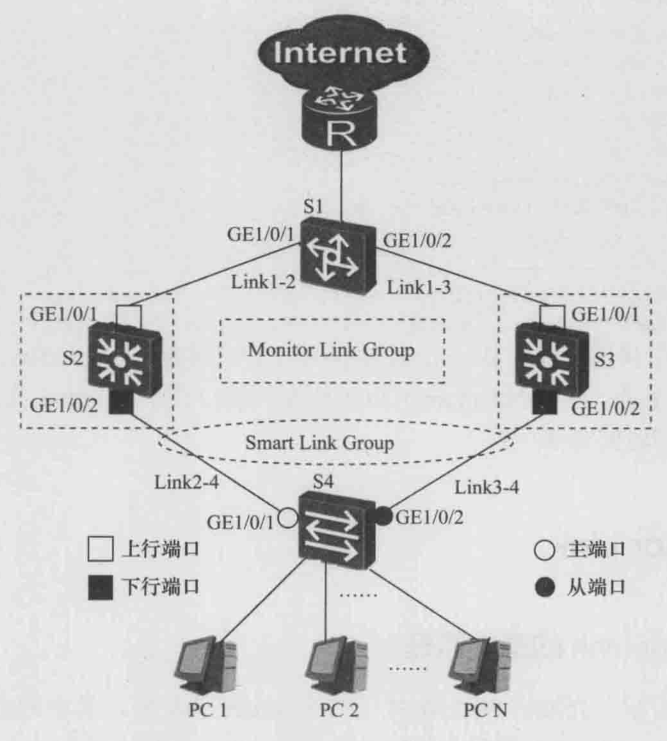


图 10-21 Monitor Link 的基本原理

我们再来看一种比较复杂的情况，如图 10-22 所示。图 10-22 中，S1、S2、S3 上分

别配置了一个 Smart Link 组，同时在 S2 和 S3 分别配置了一个 Monitor Link 组。注意，对于 S2 上的 Monitor Link 组而言，S2 上的整个 Smart Link 组才算是其上行端口，只有当该 Smart Link 组的两个端口都不能正常工作时，其下行端口才会被 Down 掉。S3 上的情况也是一样的，这里就不赘述了。

图 10-22 中，如果 S2 的主端口出现了故障，则其从端口会立即被切换到工作状态，此时，S2 上的 Monitor Link 组并不会产生联动效应。如果 S2 的主端口和从端口都出现了故障，那么 S2 的下行端口就会被 Down 掉，这就会触发 S1 上的 Smart Link 组进行切换操作。这个例子告诉我们，灵活而巧妙地将 Smart Link 技术和 Monitor Link 技术结合起来使用，往往可以很好地满足在复杂组网情况下的特殊需求。

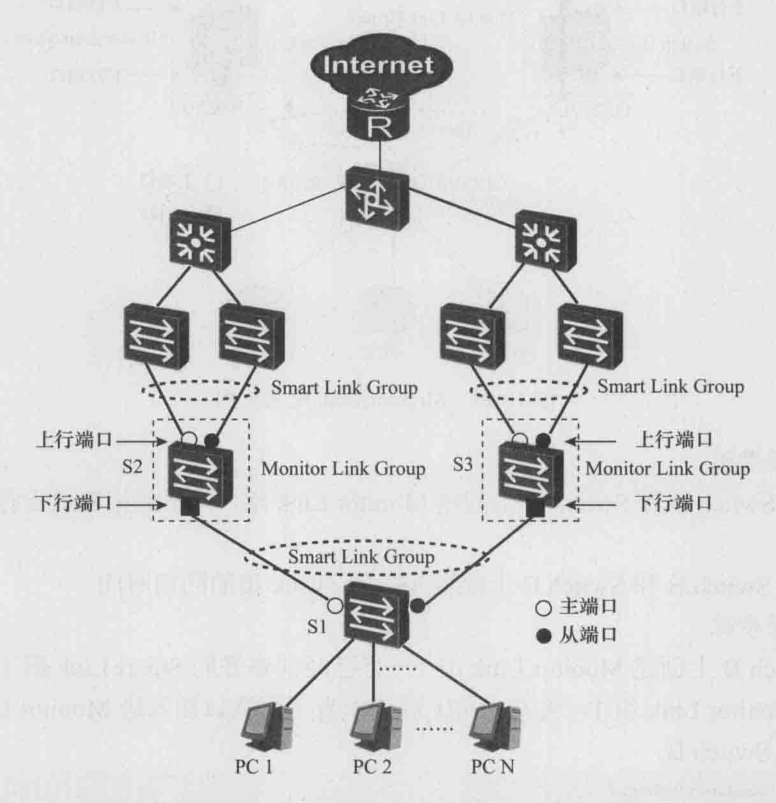


图 10-22 复杂情况下的 Monitor Link 与 Smart Link

一个 Monitor Link 组的上行端口不能正常工作时，其所有的下行端口会因此而被 Down 掉。如果上行端口恢复了正常工作，则其下行端口也会自动重新 Up，这就是 Monitor Link 的回切功能。类似于 Smart Link 的情况，我们也可以为 Monitor Link 的回切功能配置一个合适的回切时间。

10.3.2 Monitor Link 配置示例

如图 10-23 所示，Switch A 和 Switch B 上已经配置好了 Smart Link 组，我们现在需要在 Switch B 和 Switch C 上配置 Monitor Link 组。

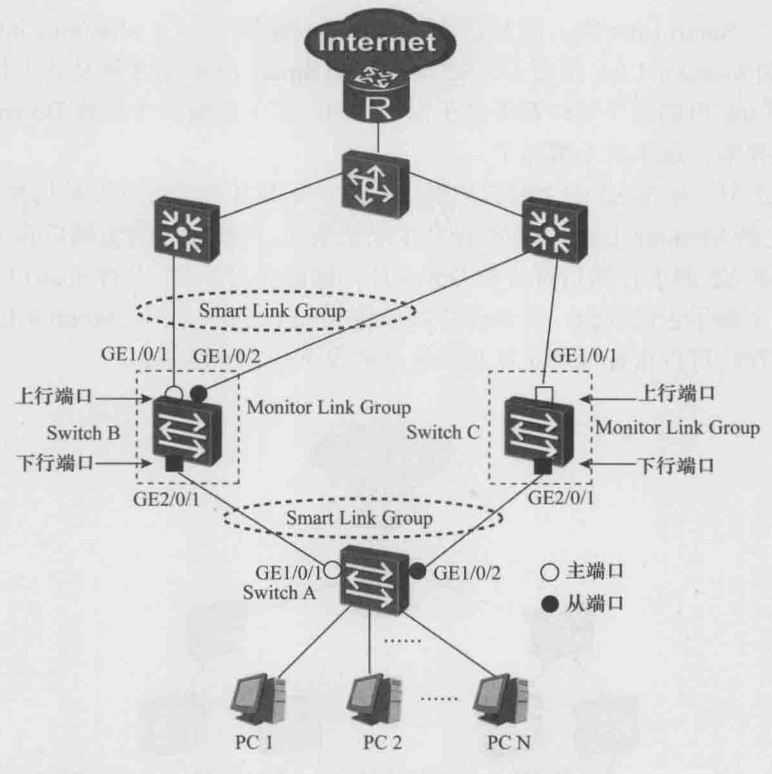


图 10-23 Monitor Link 配置示例

1. 配置思路

(1) 在 Switch B 和 Switch C 上创建 Monitor Link 组，并添加相应的上行端口和下行端口。

(2) 在 Switch B 和 Switch C 上配置 Monitor Link 组的回切时间。

2. 配置步骤

在 Switch B 上创建 Monitor Link 组 1，将已经创建好的 Smart Link 组 1 作为上行端口加入进 Monitor Link 组 1，将 GE2/0/1 端口作为下行端口加入进 Monitor Link 组 1。

配置 Switch B。

```
[SwitchB] monitor-link group 1
[SwitchB-mtlk-group1] smart-link group 1 uplink
[SwitchB-mtlk-group1] port gigabitethernet 2/0/1 downlink 1
```

在 Switch C 上创建 Monitor Link 组 2，将 GE1/0/1 端口作为上行端口加入进 Monitor Link 组 2，将 GE2/0/1 端口作为下行端口加入进 Monitor Link 组 2。

配置 Switch C。

```
[SwitchC] monitor-link group 2
[SwitchC-mtlk-group2] port gigabitethernet 1/0/1 uplink
[SwitchC-mtlk-group2] port gigabitethernet 2/0/1 downlink 1
```

然后，使用 **timer recover-time** 命令设定 Monitor Link 组的回切时间为 10 秒。

配置 Switch B。

```
[SwitchB-mtlk-group1] timer recover-time 10
```

配置 Switch C。

```
[SwitchC-mtlk-group2] timer recover-time 10
```

现在，我们需要对所做的配置进行确认，也就是使用 **display smart-link group** 命令来查看关于 Smart Link 的信息，使用 **display monitor-link group** 命令来查看关于 Monitor Link 的信息。以 Switch B 为例。

```
<SwitchB> display smart-link group 1
Smart Link group 1 information :
  Smart Link group was enabled
  Wtr-time is:   30 sec.
  There is no Load-Balance
  There is no protected-vlan reference-instance
  DeviceID:   0018-2000-0083   Control-vlan ID : 10
  Member                    Role        State        ...
  ----------------------------------------------
  GigabitEthernet1/0/1      Master      Active       ...
  GigabitEthernet1/0/2      Slave       Inactive     ...

<SwitchB> display monitor-link group 1
Monitor Link group 1 information :
  Recover-timer is 10 sec.
  Member                    Role            State   ...
 Smart-link1                UpLk            UP      ...
 GigabitEthernet2/0/1       DwLk[1] UP              ...
```

从回显信息中我们可以看到，Switch B 上的 Smart Link 组 1 已经使能，GE1/0/1 作为主端口处于 Active 状态，GE1/0/2 作为从端口处于 Inactive 状态，回切时间是 30 秒，控制 VLAN 是 VLAN 10。Monitor Link 组 1 的上行端口是 Smart Link 组 1，下行端口是 GE2/0/1，回切时间是 10 秒。

10.4　练习题

1.（多选）关于链路聚合技术，下列描述中正确的是？（　）

A．链路聚合技术可以用在两台路由器之间

B．链路聚合技术可以用在两台交换机之间

C．链路聚合技术可以用在两台服务器之间

D．链路聚合技术不可以用在一台交换机与一台路由器之间

E．链路聚合技术不可以用在一台服务器与一台路由器之间

F．链路聚合技术可以用在一台服务器与一台交换机之间

2.（多选）关于链路聚合技术，下列描述中正确的是？（　）

A．链路聚合技术可以用来灵活地增加设备之间的带宽

B．在接收端聚合端口的帧接收队列中，帧的先后顺序必须与它们在发送端聚合端口的帧发送队列中的先后顺序严格地保持一致

C．链路聚合技术可以用来增强设备之间连接的可靠性

D．Smart Link 和 LACP 都是 IEEE 针对链路聚合技术制定的标准规范

3.（单选）假设某台设备上的端口均为 GE 口，如果需要绑定出一个最大带宽可达 3.5G 的 Eth-Trunk 端口，那么至少需要将几个端口加入进这个 Eth-Trunk 端口？（　）

A. 2个　　B. 3个　　C. 4个　　D. 5个

4. （多选）关于 Smart Link 技术，下列描述中正确的是？（　）

A. 正常情况下，Smart Link 组的主端口处于 Active 状态，从端口处于 Inactive 状态

B. Smart Link 技术规范是由华为公司制定的

C. Smart Link 组的主端口和从端口必须使能 STP 功能，否则就会导致工作环路的产生

D. 如果 Smart Link 组的主端口处于 Inactive 状态，从端口处于 Active 状态，则说明 Smart Link 的配置出现了错误

5. （单选）关于 Monitor Link 技术，下列描述中正确的是？（　）

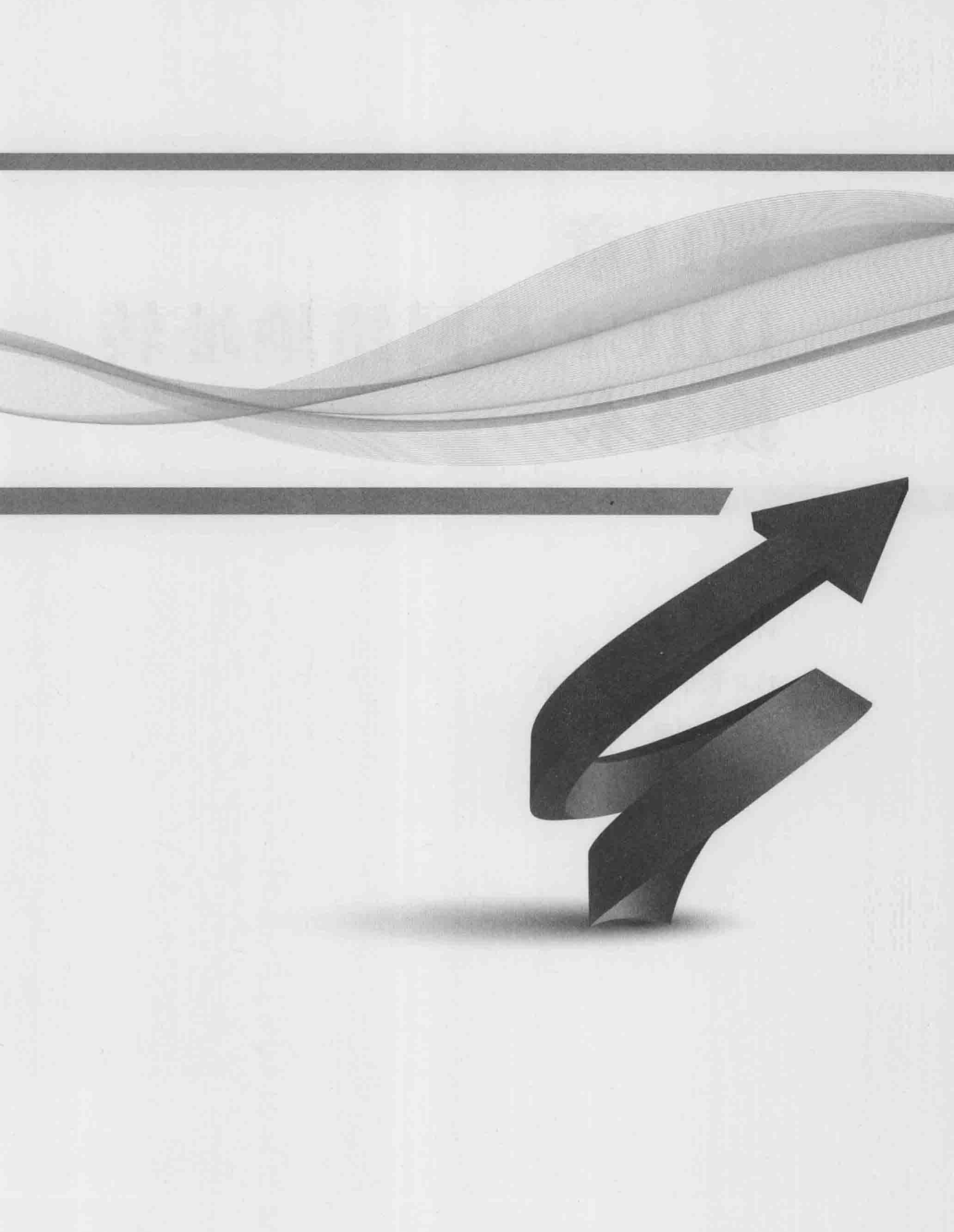
A. Monitor Link 组的上行端口的状态会随下行端口的状态变化而变化

B. Monitor Link 组只能包含一个下行端口

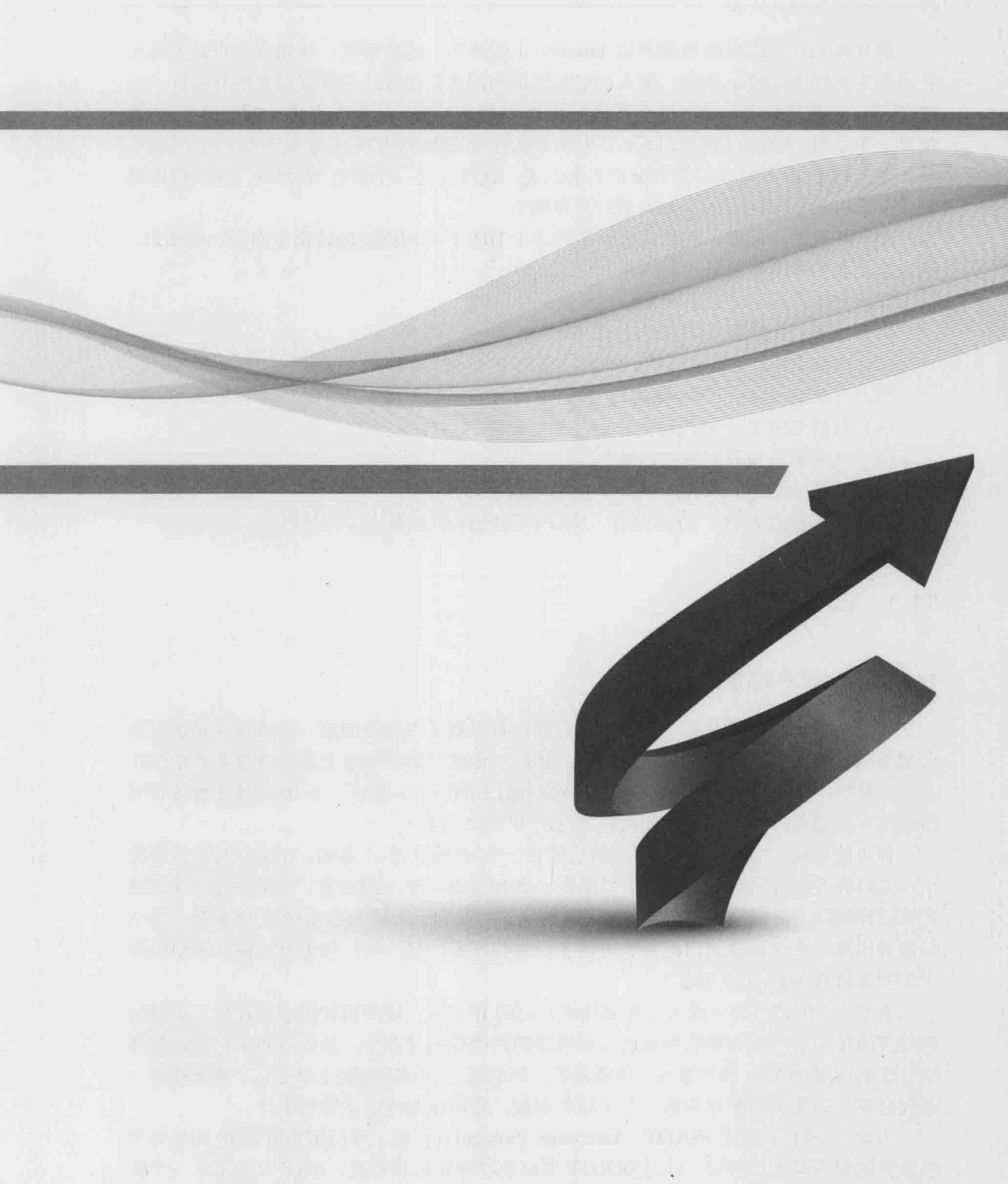
C. Smart Link 组不能作为 Monitor Link 组的上行端口

D. Monitor Link 技术规范是由 IETF 制定的

E. 以上描述都是错误的



第11章 DHCP及网络地址转换技术



假设你正在使用你的电脑浏览 Internet 上的新闻。这个时候，如果你在电脑的命令行界面下执行 ipconfig 命令，那么在电脑屏幕的回显信息中，你肯定会看到有这样一个 IP 地址，它就是你电脑的网口正在使用的 IP 地址。并且，十有八九你会发现这个 IP 地址是一个私有 IP 地址（请复习 6.4 节中关于私有 IP 地址的内容）。这就产生了两个问题，其一是，这个 IP 地址是从何而来的？其二是，既然它是一个私有 IP 地址，那么你的电脑又怎么可以与公网（Internet）进行通信呢？

要回答这两个问题，我们就必须了解关于 DHCP 及网络地址转换方面的一些知识，这也正是我们本章将要学习的内容。

学习完本章内容之后，我们应该能够：

（1）理解 DHCP 的基本概念和作用；

（2）理解 DHCP Client 首次、非首次获取 IP 地址时的工作流程；

（3）理解 IP 地址租约及租约期的概念；

（4）理解 DHCP 中继代理的作用及部署位置；

（5）理解私网与公网的基本概念；

（6）理解 NAT（网络地址转换）的基本概念和作用；

（7）理解静态 NAT、动态 NAT、NAPT 及 Easy IP 的基本工作原理。

11.1 DHCP

11.1.1 DHCP 的基本概念

设想一下，你是某个大公司的一名白领，每天到了办公室的第一件事情就是插好办公室电脑的网线和电源线，并打开电脑。然后，你就开始在电脑上聊天或是看看新闻什么的。显然，你的电脑（更准确地说，是你电脑上的网口）需要一个 IP 地址才能进行网络通信。问题是，你的电脑是如何得到这个 IP 地址的呢？

你可能会说："我可以自己手工给它配置一个 IP 地址呀。"是的，在某些特殊的情况下，你的确可以自己给你的电脑手工配置一个 IP 地址。但一般而言，你是不能够或不被允许这样做的。请想一想，如果你们公司的员工都是自己配置自己电脑的 IP 地址，那么可能会出现一些什么样的问题呢？再说了，请你好好回忆一下，你很有可能从来就没有手工配置过你电脑的 IP 地址！

事实上，你的办公电脑不仅需要知道自己的 IP 地址，还应该知道它所在的二层网络的网关地址，还应该知道它所在的二层网络的网络掩码是多少，还应该知道它附近的网络打印机的 IP 地址，如此等等。也就是说，你的办公电脑在刚刚上电之后，需要获得一系列必要的和重要的配置参数。有了这些参数，你的电脑才能正常地工作。

为此，IETF 制定了 BOOTP（Bootstrap Protocol）协议，专门用来解决 IP 地址等网络参数的配置问题。后来，针对 BOOTP 协议的各种缺陷和不足，IETF 又制定了一个新的协议，称为 DHCP（Dynamic Host Configuration Protocol），即动态主机配置协议。该协议提供了一种动态分配网络配置参数的机制，并且可以后向兼容 BOOTP 协议。

DHCP 可以分配的配置参数是多种多样的，但在本书中，我们只关注它对于主机 IP 地址的分配过程。注意，这里所说的主机（Host），是指任何需要得到 IP 地址等配置参数的网络设备，主要包括计算机（电脑）等。需要说明的是，通常情况下，路由器是不适合通过 DHCP 来自动获取它的 IP 地址的。对于路由器，我们一般应该根据它所处的网络环境及其他一些原则来手工配置它的 IP 地址等参数。

DHCP 是一种 Client/Server 模式的网络协议。需要特别说明的是，这里的 Server 虽然常常被翻译成“服务器”，但它并非是指我们平时看得见摸得着的那种服务器（高性能计算机），而只是一个应用程序而已。这个应用程序可以运行在个人电脑上，也可以运行在服务器（高性能计算机）上，还可以运行在路由器等其他设备上。同样，这里的 Client 也只是一个应用程序而已。

回到本小节第一段末尾提出的问题，“问题是，你的电脑是如何得到这个 IP 地址的呢？”简化的回答可以是这样：电脑上电之后，会自动运行 DHCP Client。DHCP Client 会与运行在公司其他设备上的 DHCP Server 进行交互，请求从 DHCP Server 那里获取自己的 IP 地址。DHCP Server 在收到 DHCP Client 的请求后，会根据某种规则在自己的地址池中选择一个 IP 地址，然后将它分配给 DHCP Client。最后，你的电脑就会把 DHCP Client 得到的 IP 地址作为你的电脑网口的 IP 地址。图 11-1 示意了 DHCP 的基本概念和作用。

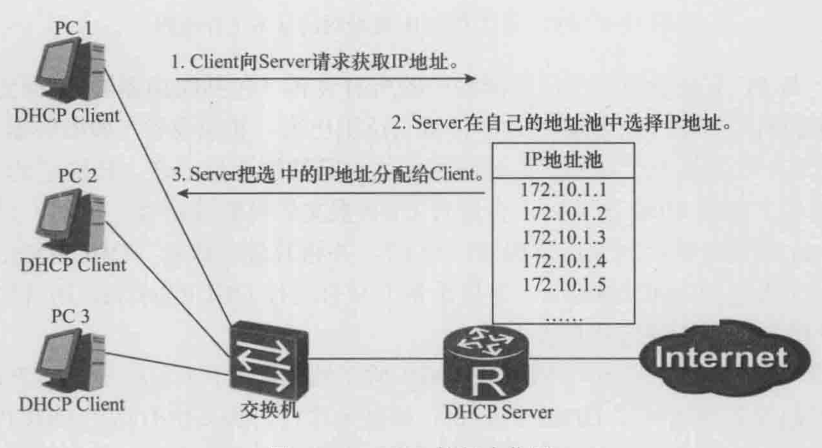


图 11-1　DHCP 的基本作用

11.1.2　DHCP 的基本工作流程

下面，我们以图 11-2 为参考，描述一下 DHCP 的基本工作流程。DHCP 的基本工作流程分为 4 个阶段，即发现阶段，提供阶段，请求阶段，确认阶段。在图 11-2 中，我们假设 PC 1 是一台刚刚买来的新电脑，这台电脑还从来没有通过 DHCP 获取过自己的 IP 地址。我们将描述 PC 1 是如何通过 DHCP 来首次获取自己的 IP 地址的。

1. 发现阶段

发现阶段也就是 PC 1 上的 DHCP Client 寻找 DHCP Server 的阶段。PC 1 上的 DHCP Client 开始运行后，会发送一个广播帧，这个广播帧的源 MAC 地址为 PC 1 的 MAC 地

址，类型字段的值为 0x0800，载荷数据为一个广播 IP 报文。该 IP 报文的目的 IP 地址为有限广播地址 255.255.255.255，源 IP 地址为 0.0.0.0（请复习 6.4 节中关于特殊 IP 地址的内容），协议字段的值为 0x11，载荷数据是一个 UDP 报文。该 UDP 报文的目的端口号为 67，源端口号为 68，载荷数据是一个 DHCPDISCOVER 消息。

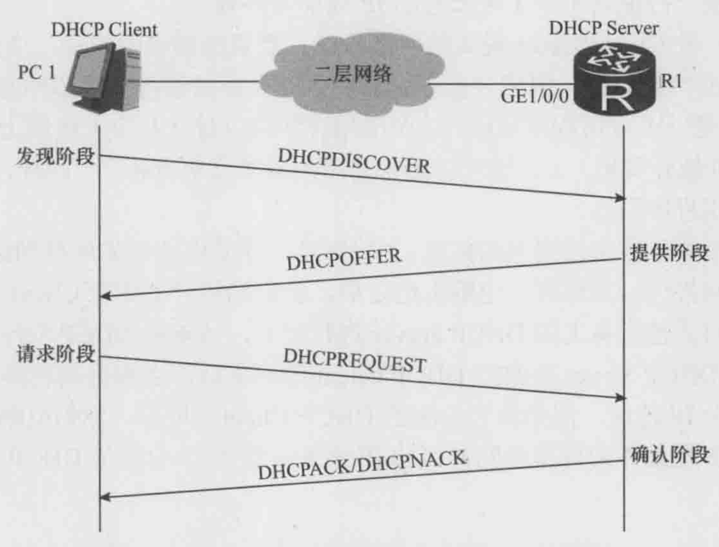


图 11-2　PC 1 首次获取 IP 地址时的基本工作流程

显然，与 PC 1 处于同一个二层网络中的所有设备（包括路由器 R1）都会收到这个广播帧。交换机收到这个广播帧后，只会将它泛洪出去。其他设备（如服务器、路由器、其他的 PC 等）收到这个广播帧后，会将相关的载荷数据逐层上送。传输层的 UDP 模块接收到网络层上送的 UDP 报文后，会检查 UDP 报文的目的端口号。显然，只有运行了 DHCP Server 的设备才会识别出目的端口号 67，并将其载荷数据（DHCPDISCOVER 消息）上送至应用层的 DHCP Server。如果设备上没有运行 DHCP Server，则目的端口号为 67 的 UDP 报文会在传输层被直接丢弃。

需要说明的是，图 11-2 所示的二层网络中除了路由器 R1 上运行了 DHCP Server 外，可能还有其他设备也运行了 DHCP Server。如果是这样，那么所有这些 DHCP Server 都会接收到 PC 1 发送的 DHCPDISCOVER 消息，也都会对所收到的 DHCPDISCOVER 消息做出回应。

从上面的描述中我们知道，DHCP 的传输层协议是 UDP，而 UDP 通信方式是一种无连接的、不那么可靠的通信方式，所以 DHCP 必须依靠自己的协议机制来提供传输的可靠性。例如，PC 1 的 DHCP Client 以广播方式发出了 DHCPDISCOVER 消息后，却没有收到任何来自 DHCP Server 的回应，那该怎么办呢？原来，DHCP 协议定义了一套消息重传机制，规定了在什么情况下需要重复发送已经发送过的消息，重复的间隔时间是多少，最大重复次数是多少，如此等等。总之，DHCP 工作过程的细节是比较复杂的，我们这里不做细究。

2. 提供阶段

提供阶段也就是 DHCP Server 向 DHCP Client 提供 IP 地址的阶段。注意，DHCP Client

是否愿意接受 DHCP Server 所提供的 IP 地址，这个阶段还反映不出来。图 11-2 中，每个接收到 DHCPDISCOVER 消息的 DHCP Server（包括路由器 R1 上运行的 DHCP Server）都会从自己维护的地址池中选择一个合适的 IP 地址，并通过 DHCPOFFER 消息将这个 IP 地址发送给 DHCP Client。

DHCPOFFER 消息是封装在目的端口号为 68、源端口号为 67 的 UDP 报文中的，该 UDP 报文又是封装在一个广播 IP 报文中的。IP 报文的目的 IP 地址为有限广播地址 255.255.255.255，源 IP 地址为 DHCP Server 所对应的单播 IP 地址，协议字段的值为 0x11。该 IP 报文又是封装在一个广播帧里的，这个帧的源 MAC 地址为 DHCP Server 所对应的单播 MAC 地址，类型字段的值为 0x0800。

显然，与 PC 1 处于同一个二层网络中的所有设备都会收到这个广播帧。交换机收到这个广播帧后，只会将它泛洪出去。其他设备（如服务器、PC 等）收到这个广播帧后，会将相关的载荷数据逐层上送。传输层的 UDP 模块接收到网络层上送的 UDP 报文后，会检查 UDP 报文的目的端口号。显然，只有运行了 DHCP Client 的设备才会识别出目的端口号 68，并将其载荷数据（DHCPOFFER 消息）上送至应用层的 DHCP Client。如果设备上没有运行 DHCP Client，则目的端口号为 68 的 UDP 报文会在传输层被直接丢弃。

现在问题来了，二层网络中除了 PC 1 外，可能还存在别的 PC，并且别的 PC 上可能也运行了 DHCP Client。那么，这些 DHCP Client 在收到 DHCPOFFER 消息后，如何才能确定这个 Offer 是不是给自己的呢？原来，每个 DHCP Client 在发送 DHCPDISCOVER 消息的时候，都会在 DHCPDISCOVER 消息中设定一个交易号（Transaction ID），DHCP Server 在回应 DHCPDISCOVER 消息的时候，会将这个交易号拷贝至 DHCPOFFER 消息中。这样一来，一个 DHCP Client 在收到一个 DHCPOFFER 消息后，只要检查其中的交易号是不是自己当初设定的交易号，就能判断出这个 Offer 是不是给自己的。顺便提一句，交易号是一个 4 字节的二进制数，所以交易号“撞车”的可能性是非常非常小的。

3. 请求阶段

在请求阶段中，PC 1 的 DHCP Client 会在若干个收到的 Offer（即若干个收到的 DHCPOFFER 消息）中根据某种原则来确定出自己将要接受哪一个 Offer。通常情况下，DHCP Client 会接受它所收到的第一个 Offer（即最先收到的那个 DHCPOFFER 消息）。图 11-2 中，假设 PC 1 最先收到的 DHCPOFFER 消息是来自路由器 R1。于是，PC 1 的 DHCP Client 会发送一个广播帧，这个广播帧的意图就是向路由器 R1 上的 DHCP Server 提出请求，希望获取到该 DHCP Server 发送给自己的 DHCPOFFER 消息中所提供的那个 IP 地址。

PC 1 的 DHCP Client 发送的广播帧的源 MAC 地址为 PC 1 的 MAC 地址，类型字段的值为 0x0800，载荷数据是一个广播 IP 报文。该 IP 报文的目的 IP 地址为有限广播地址 255.255.255.255，源 IP 地址为 0.0.0.0，协议字段的值为 0x11，载荷数据是一个 UDP 报文。该 UDP 报文的目的端口号为 67，源端口号为 68，载荷数据是一个 DHCPREQUEST 消息。注意，这个 DHCPREQUEST 消息中携带有 R1 上的 DHCP Server 的标识（称为 Server Identifier），表示 PC 1 的 DHCP Client 只愿意接受 R1 上的 DHCP Server 所给出的 Offer。

显然，该二层网络上所有的 DHCP Server 都会接收到 PC 1 上的 DHCP Client 发送的 DHCPREQUEST 消息。R1 上的 DHCP Server 收到并分析了该 DHCPREQUEST 消息后，会明白 PC 1 已经愿意接受自己的 Offer 了。其他的 DHCP Server 收到并分析了该

DHCPREQUEST 消息后，会明白 PC 1 拒绝了自己的 Offer。于是，这些 DHCP Server 就会收回自己当初给予 PC 1 的 Offer。也就是说，当初准备提供给 PC 1 使用的 IP 地址现在可以用来分配给别的设备使用了。

4. 确认阶段

在确认阶段，R1 上的 DHCP Server 会向 PC 1 上的 DHCP Client 发送一个 DHCPACK 消息。DHCPACK 消息是封装在目的端口号为 68、源端口号为 67 的 UDP 报文中的，该 UDP 报文又是封装在一个广播 IP 报文中的。IP 报文的目的 IP 地址为有限广播地址 255.255.255.255，源 IP 地址为 DHCP Server 所对应的单播 IP 地址，协议字段的值为 0x11。注意，该 IP 报文是封装在一个单播帧里的，这个帧的源 MAC 地址为 DHCP Server 所对应的单播 MAC 地址，目的 MAC 地址为 PC 1 的 MAC 地址，类型字段的值为 0x0800。注意，由于种种原因，R1 上的 DHCP Server 也可能会向 PC 1 上的 DHCP Client 发送一个 DHCPNACK 消息。如果 PC 1 接收到了 DHCPNACK 消息，就说明这次获取 IP 地址的尝试失败了。在这种情况下，PC 1 只能重新回到发现阶段来开始新一轮的 IP 地址申请过程。

PC 1 上的 DHCP Client 接收到 R1 上的 DHCP Server 发送的 DHCPACK 消息后，就意味着 PC 1 首次获得了 DHCP Server 分配给自己 IP 地址。实际上，PC 1 还会立即通过 Gratuitous ARP 机制来检验所获得的 IP 地址的唯一性，但这个过程我们这里就不描述了。

我们不禁要问，PC 1 下一次开机启动的时候，是否也需要完全重复前面所述的 4 个阶段（发现阶段，提供阶段，请求阶段，确认阶段）才能获得 IP 地址呢？答案是否定的，如图 11-3 所示。事实上，PC 1 上是有磁盘等存储设备的，因此 PC 1 是能够记住自己上次所获得的 IP 地址的，并且也能记住当初分配这个 IP 地址的那个 DHCP Server 的 Server Identifier（也就是 R1 上的 DHCP Server 的 Server Identifier），还能记住这个 DHCP Server 所对应的单播 IP 地址和单播 MAC 地址等信息。所以，PC 1 重新启动的时候，只需要直接进入第 3 个阶段（请求阶段），以广播帧及广播 IP 报文的方式发送 DHCPREQUEST 消息（该消息中携带有 R1 上的 DHCP Server 的 Server Identifier），表示希望继续使用上次分配给自己的 IP 地址。PC 1 在收到来自 R1 上的 DHCP Server 的 DHCPACK 消息后，就又可以开始继续使用原来的那个 IP 地址了。如果由于种种原因，R1 上的 DHCP Server 不能让 PC 1 继续使用这个 IP 地址，那么 R1 上的 DHCP Server 就会回应一个 DHCPNACK 消息。PC 1 如果收到了 DHCPNACK 消息，就必须放弃使用原来的 IP 地址，而必须重新从发现阶段开始来重新申请一个 IP 地址。

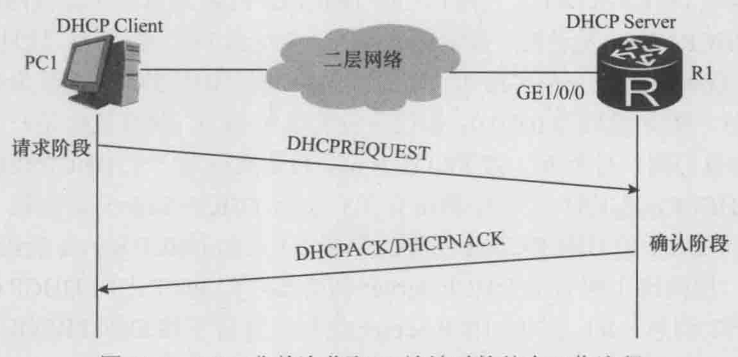


图 11-3　PC 1 非首次获取 IP 地址时的基本工作流程

细心的读者可能会问，既然 PC 1 记住了 R1 上的 DHCP Server 所对应的单播 IP 地址和单播 MAC 地址（也就是 R1 的 GE1/0/0 接口的 IP 地址和 MAC 地址），那么，PC 1 重新启动的时候，为何是以广播帧及广播 IP 报文的方式发送 DHCPREQUEST 消息，而不是以“影响面较小的”单播帧及单播 IP 报文的方式来发送 DHCPREQUEST 消息呢？事实上，DHCP 协议是允许先以单播帧及单播 IP 报文的方式来发送 DHCPREQUEST 消息的，如果发送之后接收不到回应（例如，R1 的 GE1/0/0 接口的 IP 地址或 MAC 地址发生了改变），那么就再以广播帧及广播 IP 报文的方式发送 DHCPREQUEST 消息。

从 DHCP 协议的角度来看，IP 地址的所有权是属于 DHCP Server 的，而不是 DHCP Client 的；DHCP Client 所拥有的只是 IP 地址的使用权。事实上，DHCP Server 每次给 DHCP Client 分配一个 IP 地址时，只是跟 DHCP Client 订立了一个关于这个 IP 地址的租约（Lease）。每个租约都有一个租约期（Duration of Lease），DHCP 协议规定租约期的缺省值不得小于 1 个小时，而实际部署 DHCP 时，租约期的缺省值通常都是 24 小时。在租约期内，DHCP Client 才能使用相应的 IP 地址。当租约期到期之后，DHCP Client 是不被允许继续使用这个 IP 地址的。当然了，在租约期还没有到期的时候，DHCP Client 是可以申请续租这个 IP 地址的，申请的流程如图 11-4 所示。

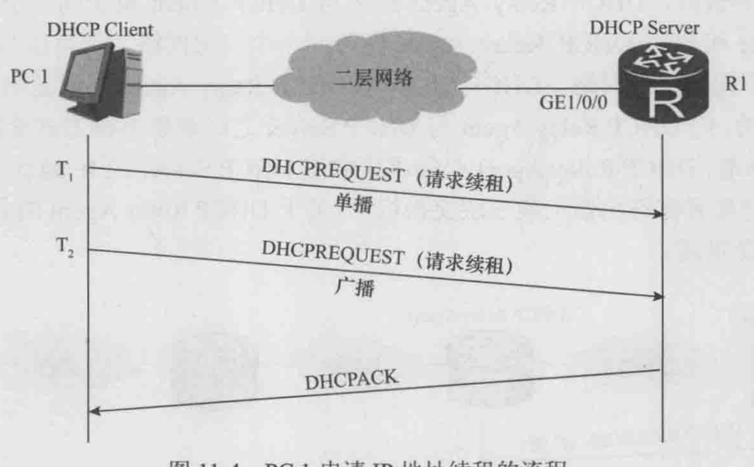


图 11-4　PC 1 申请 IP 地址续租的流程

按照 DHCP 协议的规定，在缺省情况下，图 11-4 中的 T_1 时刻是租约期到了一半的时刻，而 T2 时刻则是租约期到了 87.5%的时刻。在 T_1 时刻，PC 1 上的 DHCP Client 会以单播方式向 R1 上的 DHCP Server 发送一个 DHCPREQUEST 消息，请求续租 IP 地址（也就是请求重新开始租约期的计时）。如果在 T_2 时刻之前，PC 1 上的 DHCP Client 收到了回应的 DHCPACK 消息，则说明续租已经成功。如果直到 T_2 时刻，PC 1 上的 DHCP Client 都未收到回应的 DHCPACK 消息，那么在 T_2 时刻，PC 1 上的 DHCP Client 会以广播方式发送一个 DHCPREQUEST 消息，继续请求续租 IP 地址。如果在租约期到期之前，PC 1 上的 DHCP Client 收到了回应的 DHCPACK 消息，则说明续租成功。如果直到租约期到期时，PC 1 上的 DHCP Client 仍未收到回应的 DHCPACK 消息，那么 PC 1 就必须停止使用原来的 IP 地址，也就是说，PC 1 只能重新从发现阶段开始来重新申请一个 IP 地址。

11.1.3 DHCP 中继代理

仔细回忆一下上一小节中我们所描述的 DHCP 基本工作流程就会发现，DHCP Client 总是以广播（广播帧及广播 IP 报文）方式来发送 DHCPDISCOVER 消息和 DHCPREQUEST 消息的。如果 DHCP Server 和 DHCP Client 不在同一个二层网络（二层广播域）中，那么 DHCP Server 根本就不可能接收到这些 DHCPDISCOVER 消息和 DHCPREQUEST 消息。因此，我们之前所描述的 DHCP 工作流程，只适合于 DHCP Server 和 DHCP Client 位于同一个二层网络的场景。

如果一个公司的网络包含了多个二层网络，那么我们是不是必须在每个二层网络中都至少部署一个 DHCP Server 呢？从理论上讲，这样做未尝不可。但实际上，这样做是没有必要的，也是很不经济的。事实上，DHCP 协议除了定义了 DHCP Client 和 DHCP Server 这两种角色之外，还定义了 DHCP Relay Agent（DHCP 中继代理）这种角色。DHCP Relay Agent 的基本作用就是专门在 DHCP Client 和 DHCP Server 之间进行 DHCP 消息的中转。

如图 11-5 所示，DHCP Client 利用 DHCP Relay Agent 来从 DHCP Server 那里获取 IP 地址等配置参数时，DHCP Relay Agent 必须与 DHCP Client 位于同一个二层网络，但 DHCP Server 可以与 DHCP Relay Agent 位于同一个二层网络，也可以与 DHCP Relay Agent 位于不同的二层网络。DHCP Client 与 DHCP Relay Agent 之间是以广播方式交换 DHCP 消息的，但 DHCP Relay Agent 与 DHCP Server 之间是以单播方式交换 DHCP 消息的（这就意味着，DHCP Relay Agent 必须事先知道 DHCP Server 的 IP 地址）。DHCP Relay Agent 通常是部署在路由器上或三层交换机上。关于 DHCP Relay Agent 的具体工作原理，我们这里不做描述。

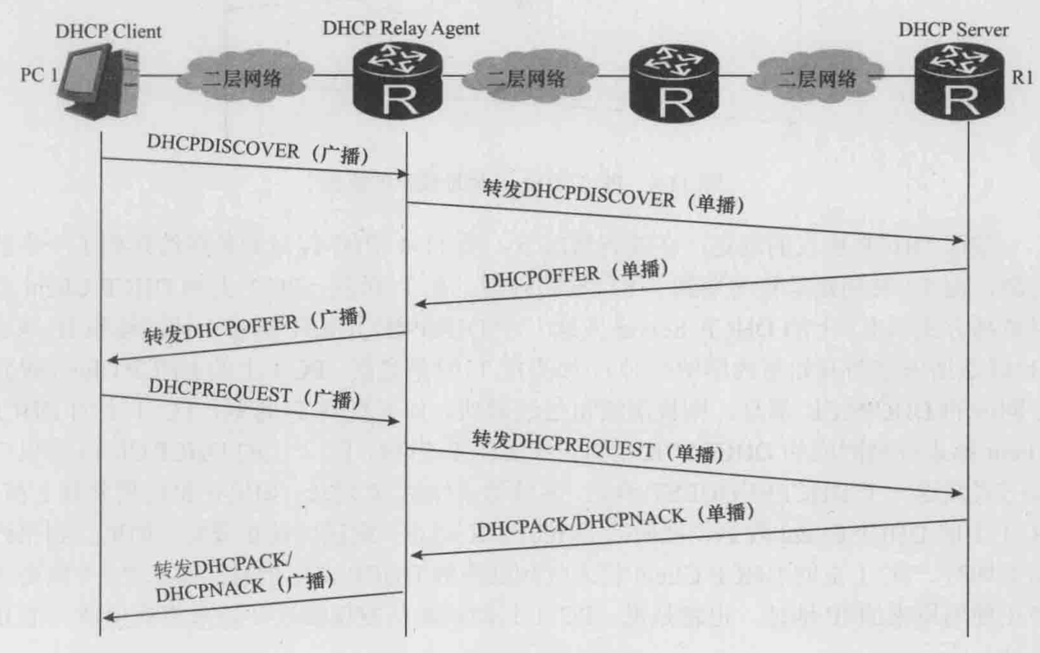


图 11-5 DHCP 中继代理

11.1.4　DHCP Server 配置示例

如图 11-6 所示，某公司有 3 个部门，不同的部门位于不同的网段（二层网络），每个部门的 PC 数目在 50 台左右，并且不会超过 60 台。现在需要在路由器 R1 上配置 DHCP Server，以便给各个部门的 PC 提供 DHCP 配置服务。

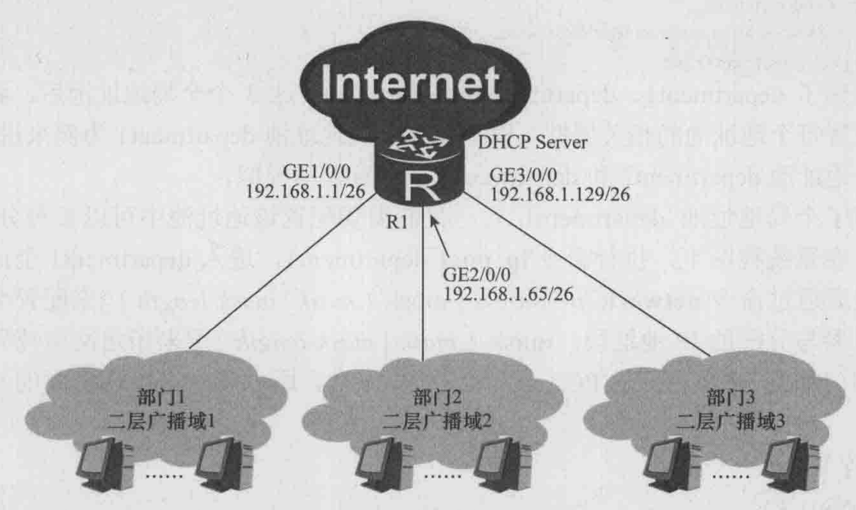


图 11-6　DHCP Server 配置示例

1. 配置思路

（1）在 R1 上使能 DHCP 功能。

（2）创建三个全局地址池，用于为三个不同部门的 PC 分配 IP 地址。

（3）配置地址池的相关属性。

（4）在 R1 的接口下配置基于全局地址池的服务方式，实现 DHCP Server 从全局地址池中选择分配 IP 地址。

2. 配置步骤

要在 R1 上使能 DHCP 功能，必须先进入系统视图，然后执行命令 **dhcp enable**。

#配置 R1。

```
<R1> system-view
[R1] dhcp enable
```

然后，我们可以在 R1 上创建 3 个不同的全局地址池，分别用于为 3 个不同部门的 PC 分配 IP 地址。DHCP Server 分配 IP 地址时，可以使用基于全局地址池的服务方式，也可以使用基于接口地址池的服务方式。一般情况下，我们通常使用基于全局地址池的服务方式，因为在这种方式下，不必要求 DHCP Client 和 DHCP Server 位于同一个二层网络。在基于接口地址池的服务方式下，只有当 DHCP Client 与该接口位于同一个二层网络时，DHCP Client 才可以从该接口地址池中获取 IP 地址。

创建全局地址池的命令是 **ip pool** *ip-pool-name*，*ip-pool-name* 表示地址池名称，它可以是字母、数字、下划线等符号的组合。

#配置 R1。

```
<R1> system-view
[R1] ip pool department1
Info:  It's successful to create an IP address pool.
[R1-ip-pool-department1] quit
[R1] ip pool department2
Info:  It's successful to create an IP address pool.
[R1-ip-pool-department2] quit
[R1] ip pool department3
Info:  It's successful to create an IP address pool.
[R1-ip-pool-department3] quit
```

在创建了 department1、department2、department3 这 3 个全局地址池后，我们还需要逐一配置每个地址池的相关属性。以下仅以全局地址池 department1 为例来进行示意，其他两个地址池 department2 和 department3 的配置非常类似。

创建了全局地址池 department1 后，我们需要配置该地址池中可以参与分配的 IP 地址段。在系统视图下，执行命令 **ip pool** department1，进入 department1 全局地址池视图，然后通过命令 **network** *ip-address* [**mask** { *mask* | *mask-length* }]来配置全局地址池中可以参与分配的 IP 地址段。**mask** { *mask* | *mask-length* }用来指定网络掩码，考虑到每个部门都有 50 台左右的 PC，且不会超过 60 台，所以我们可以将掩码的长度确定为 26。

#配置 R1。

```
<R1> system-view
[R1] ip pool department1
[R1-ip-pool-department1] network 192.168.1.0 mask 26
```

这样一来，department1 地址池中可以参与分配的 IP 地址就是 192.168.1.1~192.168.1.62（一共包含了 62 个 IP 地址）；192.168.1.0 和 192.168.1.63 这两个地址是不能参与分配的（请想一想，它们为何不能参与分配？）。

接下来，我们将配置 DHCP Server 全局地址池中分配给网关的 IP 地址。命令 **gateway-list** *ip-address* 可以用来指定 PC 在获取到 IP 地址后进行网络通信时应该使用的网关 IP 地址。

#配置 R1。

```
[R1-ip-pool-department1] gateway-list 192.168.1.1
```

网关地址 192.168.1.1 配置以后，系统将自动保留该地址，不会再将该地址分配出去用作他用。细心的读者可能已经看出，192.168.1.1 这个地址其实就是 R1 的 GE1/0/0 接口的 IP 地址，而 R1 的 GE1/0/0 接口正是部门 1 的网关。

接下来，我们可以使用命令 **lease**{ **day** *day* [**hour** *hour* [**minute** *minute*]] | **unlimited** } 来配置 IP 地址的租约期。默认情况下，租约期是 1 天，我们这里将它配置成 1 个小时。

#配置 R1。

```
[R1-ip-pool-department1] lease day 0 hour 1
```

至此，我们就完成了关于地址池 department1 的配置工作。下面给出关于地址池 department2 和 department3 的配置。

#配置 R1。

```
<R1> system-view
[R1] ip pool department2
[R1-ip-pool-department2] network 192.168.1.64 mask 26
```

```
[R1-ip-pool-department2] gateway-list 192.168.1.65
[R1-ip-pool-department2] lease day 0 hour 1
[R1-ip-pool-department2] quit
[R1] ip pool department3
[R1-ip-pool-department3] network 192.168.1.128 mask 26
[R1-ip-pool-department3] gateway-list 192.168.1.129
[R1-ip-pool-department3] lease day 0 hour 1
```

现在，我们需要在 R1 的各个接口下配置基于全局地址池的服务方式，使用的命令是 **dhcp select global**。

#配置 R1。

```
<R1> system-view
[R1] interface   GigabitEthernet   1/0/0
[R1-GigabitEthernet1/0/0] ip address 192.168.1.1 26
[R1-GigabitEthernet1/0/0] dhcp select global
[R1-GigabitEthernet1/0/0] quit
[R1] interface   GigabitEthernet   2/0/0
[R1-GigabitEthernet2/0/0] ip address 192.168.1.65 26
[R1-GigabitEthernet2/0/0] dhcp select global
[R1-GigabitEthernet2/0/0] quit
[R1] interface   GigabitEthernet   3/0/0
[R1-GigabitEthernet3/0/0] ip address 192.168.1.129 26
[R1-GigabitEthernet3/0/0] dhcp select global
[R1-GigabitEthernet3/0/0] quit
```

为了验证所做的配置，我们可以在 R1 的用户视图下执行命令 **display ip pool name** *pool-name* **used**，查看地址池的配置情况以及已经分配的地址情况。

```
<R1> display ip pool name department1 used
  Pool-name                  : department1
  Pool-No                    : 0
  Lease                      : 0 Days 1 Hours 0 Minutes
  Domain-name                : -
  DNS-server0                : -
  NBNS-server0               : -
  Netbios-type               : -
  Position                   : Local          Status            : Unlocked
  Gateway-0                  : 192.168.1.1
  Mask                       : 255.255.255.192
  VPN instance               : --
-----------------------------------------------------------------
Start            End              Total    Used   Idle（Expired） Conflict    Disable
-----------------------------------------------------------------
192.168.1.1      192.168.1.62     61       1      60（0）          0           0
-----------------------------------------------------------------
Network section :
-----------------------------------------------------------------
   Index            IP              MAC              Lease    Status
-----------------------------------------------------------------
      2           192.168.1.3      000B-09CF-B353     60       used
-----------------------------------------------------------------
```

可以看到，回显信息的内容与我们的预期是一致的。另外，从回显信息中我们还可以看到，IP 地址 192.168.1.3 已经分配给了某台 PC 使用，该 PC 的 MAC 地址是 000B-09CF-B353。

11.1.5 DHCP 中继代理配置示例

图 11-6 中，我们在路由器 R1 的旁边侧挂一台服务器，同时将 R1 上的 DHCP Server 移至服务器上，并在 R1 上配置 DHCP Relay Agent，这样便得到了图 11-7 所示的网络。下面就简单介绍一下应该如何配置 R1 上的 DHCP Relay Agent。

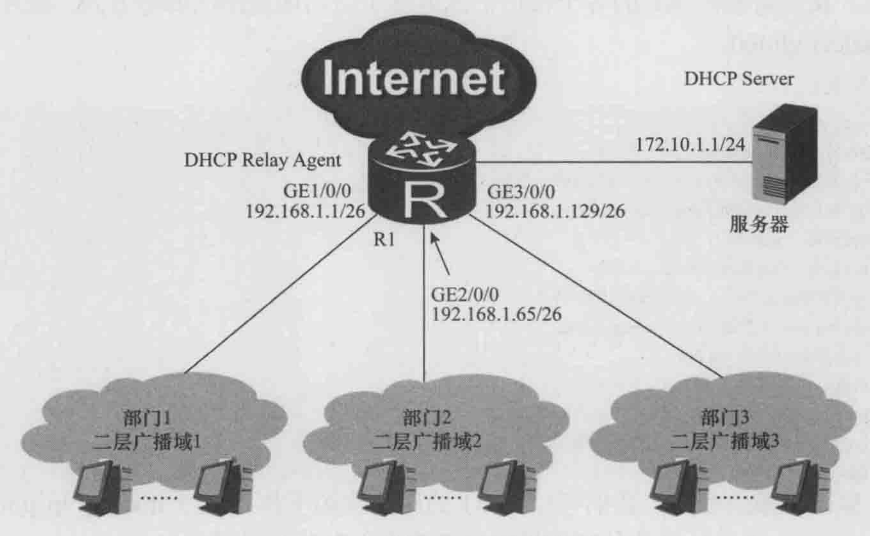


图 11-7 配置 DHCP 中继代理

1. 配置思路

（1）在 R1 上使能 DHCP 功能。

（2）在 R1 的各个接口下使能 DHCP 中继功能，并配置 DHCP Server 的 IP 地址。

2. 配置步骤

要在 R1 上配置 DHCP 中继服务，必须首先进入系统视图，然后执行命令 **dhcp enable**，使能 DHCP 功能。

#配置 R1。

```
<R1> system-view
[R1] dhcp enable
```

接下来我们在 R1 的各个接口下使能 DHCP 中继功能，使用的命令是 **dhcp select relay**。命令 **dhcp relay server-ip** 172.10.1.1 用来配置 DHCP Server 的 IP 地址。

#配置 R1。

```
[R1] interface GigabitEthernet 1/0/0
[R1-GigabitEthernet1/0/0] ip address 192.168.1.1 26
[R1-GigabitEthernet1/0/0] dhcp select relay
[R1-GigabitEthernet1/0/0] dhcp relay server-ip 172.10.1.1
[R1-GigabitEthernet1/0/0] quit
[R1] interface GigabitEthernet 2/0/0
[R1-GigabitEthernet2/0/0] ip address 192.168.1.65 26
[R1-GigabitEthernet2/0/0] dhcp select relay
[R1-GigabitEthernet2/0/0] dhcp relay server-ip 172.10.1.1
[R1-GigabitEthernet2/0/0] quit
[R1] interface GigabitEthernet 3/0/0
```

```
[R1-GigabitEthernet3/0/0] ip address   192.168.1.129 26
[R1-GigabitEthernet3/0/0] dhcp select relay
[R1-GigabitEthernet3/0/0] dhcp relay server-ip 172.10.1.1
[R1-GigabitEthernet3/0/0] quit
```

为了验证所做的配置，我们可以在 R1 的接口下使用 **display this** 命令来查看相关的 DHCP 中继配置。以 R1 的 GE1/0/0 接口为例。

```
[R1] interface GigabitEthernet 1/0/0
[R1-GigabitEthernet1/0/0] display this
#
interface GigabitEthernet1/0/0
                    ip address 192.168.1.0 255.255.255.192
                    dhcp select relay
                    dhcp relay server-ip 172.10.1.1
                    #
                    return
```

可以看到，回显信息的内容与我们的预期是一致的。

11.2　网络地址转换技术

11.2.1　网络地址转换技术的基本概念

网络地址转换技术也称为 NAT（Network Address Translation）技术，它的基本作用就是实现私网 IP 地址与公网 IP 地址之间的转换。

在 IP 地址的空间里，A、B、C 三类地址中各有一部分地址，它们被称为私网 IP 地址（或私有 IP 地址），内容如下。

（1）A 类：10.0.0.0 ~ 10.255.255.255。

（2）B 类：172.16.0.0 ~ 172.31.255.255。

（3）C 类：192.168.0.0 ~ 192.168.255.255。

除了私网 IP 地址外，IP 地址空间里的其他地址都称为公网 IP 地址（或公有 IP 地址）。由于 IP 地址的公、私属性的不同，我们便有了公网的概念和私网的概念之分。所谓公网，就是使用公有 IP 地址的网络，公网中是绝对不能使用私有 IP 地址的。在公网中，各个网络接口的 IP 地址必须是公有 IP 地址，另外，公网中出现的 IP 报文，其目的 IP 地址和源 IP 地址也都必须是公有 IP 地址。所谓私网，就是使用私有 IP 地址的网络。在私网中，各个网络接口的 IP 地址必须是私有 IP 地址，但在有些情况下，私网中出现的 IP 报文，其目的 IP 地址或源 IP 地址可以是公有 IP 地址。我们这里不去探究那些特殊 IP 地址的公、私属性。例如，我们不去探究 0.0.0.0 这个特殊 IP 地址的公、私属性，事实上，0.0.0.0 这个 IP 地址在公网中和私网中都是可以出现的。另外，我们也不去探究组播 IP 地址的公、私属性。例如，我们不去探究 224.0.0.9 这个组播 IP 地址的公、私属性，事实上，224.0.0.9 这个组播 IP 地址在公网中和私网中都是可以出现的。

再来说说 Internet。Internet 这个术语的含义，常常会因为上下文意思的不同而有所不同。如图 11-8 所示，广义地讲，私网也是 Internet 的组成部分，也就是说，Internet 包

含了公网和私网两大部分。狭义地讲，私网不算是 Internet 的组成部分。在谈及 NAT 技术时，我们所说的 Internet 总是取其狭义的含义。也就说，在谈及 NAT 技术时，公网就是指 Internet，Internet 就是指公网。



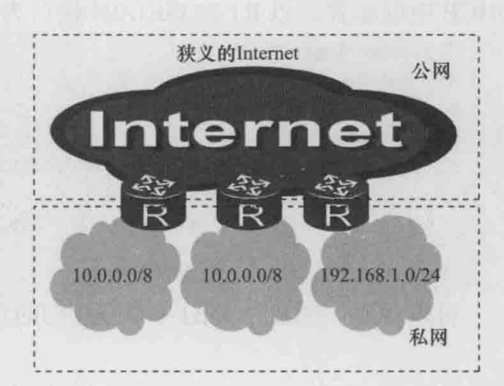


图 11-8　Internet 的范围

凡是 Internet（公网）上的网络设备，均不会接收、发送或者转发源 IP 地址或目的 IP 地址为私网 IP 地址的 IP 报文。简而言之，私网 IP 地址是不能出现在 Internet 上的。另外，在 Internet 上，IP 地址还需要满足唯一性的要求。

在同一个私网中，私有 IP 地址也需要满足唯一性的要求。然而，在不同的私网中，私有 IP 地址则无需满足唯一性的要求。例如，在图 11-8 中，两个不同的私网都使用了 10.0.0.0/8 这个网段的 IP 地址。私网 IP 地址的这种可重复使用的特点，使得私网的建设得到了充分自由的发展。

私有 IP 地址的可重用性，极大地缓解了 IP 地址资源枯竭的问题。我们知道，IP 地址的长度是 32bit，IP 地址空间总共只包含了大约 43 亿个 IP 地址（世界人口已超过了 70 亿，平均每个人还分不到 1 个 IP 地址）。这 43 亿个 IP 地址对于现今的网络发展需求来说是远远不够的。如果没有私有 IP 地址的运用以及私网的大量建设，网络技术的发展和应用或许早就因 IP 地址的枯竭问题而停滞不前了。

为了实现私网与 Internet 之间的通信，以及通过 Internet 实现私网与私网之间的通信，人们便引入了 NAT 技术，如图 11-9 所示。

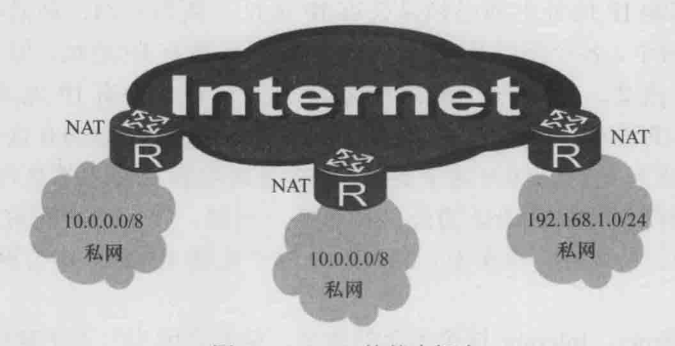


图 11-9　NAT 的基本概念

NAT 本身是一个非常宽泛的概念，具体的 NAT 技术种类及其相应的适用场景是非常繁杂的。例如，某些 NAT 技术只能适合于私网方面向公网方面发起通信的场景，反之则不行。因此，在实际部署 NAT 技术时，必须仔细地分析具体的网络环境及网络需求。在接下来的几个小节中，我们将通过举例的方式来简单地介绍一下几种基本的 NAT 技术的概念和原理。所有的例子都假设了这样一个前提：在私网与公网进行通信时，发起通信的一方总是私网。

11.2.2 静态 NAT

我们先来看看一种最为简单的 NAT 技术，称为静态 NAT，也称为简单 NAT（Simple NAT）。

如图 11-10 所示，某公司有一个私有网络，该私有网络通过路由器 R2 与 Internet 相连。R2 的 GE2/0/0 接口一侧是 Internet，GE1/0/0 接口一侧是私网。私网包含了两个网段（即两个二层网络），分别是 192.168.1.0/24 和 192.168.2.0/24。私网中共使用了 7 个私有 IP 地址（192.168.1.1，192.168.1.2，192.168.1.3，192.168.1.4，192.168.2.1，192.168.2.2，192.168.2.3），同时，该公司获得了 7 个可用的公有 IP 地址 200.24.5.1~200.24.5.7。

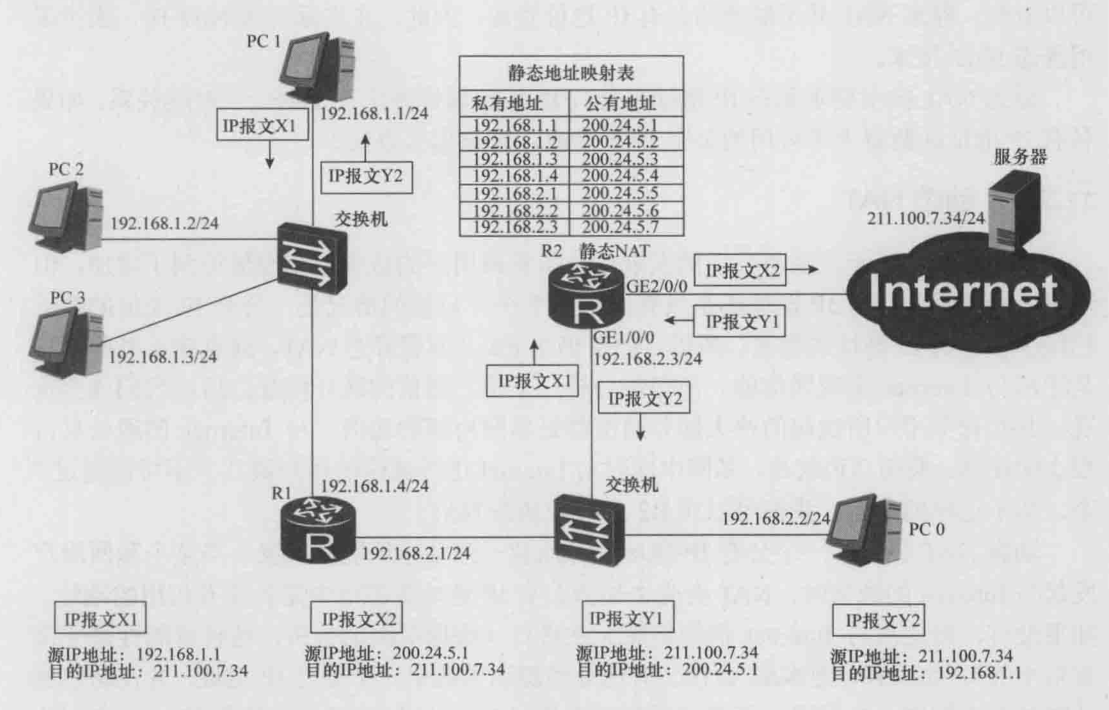


图 11-10 静态 NAT

为了能够实现私网与 Internet 之间的通信，我们可以在 R2 上部署静态 NAT。静态 NAT 技术的核心内容就是建立并维护一张静态地址映射表，如图 11-10 所示。静态地址映射表反映了公有 IP 地址与私有 IP 地址之间的一一对应关系。

图 11-10 中，假设 PC 1 向 Internet 中的服务器发起了通信，也就是 PC 1 向服务器

发送了一个 IP 报文 X1。显然，X1 的源 IP 地址为私有 IP 地址 192.168.1.1，目的 IP 地址为公有 IP 地址 211.100.7.34。当 X1 到达 R2 之后，NAT 会检查 X1 的目的 IP 地址是不是公有 IP 地址。如果是，就会在静态地址映射表中去查找 X1 的源 IP 地址所对应的公有 IP 地址。图 11-10 中的静态地址映射表表明，私有 IP 地址 192.168.1.1 所对应的公有 IP 地址是 200.24.5.1。于是，NAT 会将 X1 的源 IP 地址 192.168.1.1 替换为公有 IP 地址 200.24.5.1（注意，由于 X1 的源 IP 地址发生了改变，所以 X1 的校验和字段的值等内容也必须进行相应的改变。对于这些 NAT 技术的细节问题，我们不去分析），从而得到一个新的 IP 报文 X2。X2 会通过 R2 的 GE2/0/0 接口去往 Internet，并最终到达服务器。

当服务器向 PC 1 返回一个 IP 报文 Y1 时，Y1 的源 IP 地址应为公有 IP 地址 211.100.7.34，目的 IP 地址应为公有 IP 地址 200.24.5.1。Y1 进入 R2 后，NAT 会在静态地址映射表中查找 Y1 的目的 IP 地址 200.24.5.1，发现其对应的私有 IP 地址为 192.168.1.1。然后，NAT 会将 Y1 的目的 IP 地址 200.24.5.1 替换为私有 IP 地址 192.168.1.1，从而得到一个新的 IP 报文 Y2。Y2 会通过 R2 的 GE1/0/0 接口去往私网，并最终到达 PC 1。

从上面的描述中我们可以看到，静态 NAT 的工作原理是非常简单的。但同时我们也可以看到，静态 NAT 并不能节约公有 IP 地址资源。因此，在实际部署 NAT 时，很少采用静态 NAT 技术。

静态 NAT 技术要求私有 IP 地址与公有 IP 地址保持固定不变的一一对应关系。如果私有 IP 地址的数量大于可用的公有 IP 地址时，那该怎么办呢？

11.2.3 动态 NAT

如图 11-11 所示，随着公司的发展，公司私网用户的数量也相应地得到了增加，但可供公司使用的公有 IP 地址还是只有原来那 7 个。目前的情况是，公有 IP 地址的数量已经少于私有 IP 地址的数量。所以，如果仍在 R2 上部署静态 NAT，就意味着某些用户是无法与 Internet 实现通信的。但仔细分析一下用户通信的统计特征之后，我们就会发现，其实每个用户所发起的绝大部分通信都是私网内部的通信，与 Internet 的通信只占极少的比例。换句话说就是，私网中同时与 Internet 进行通信的用户数几乎不可能超过 7 个。基于这样的分析，我们可以在 R2 上部署动态 NAT。

动态 NAT 包含了一个公有 IP 地址资源池和一张动态地址映射表。当某个私网用户发起与 Internet 的通信时，NAT 会先去检查公有 IP 地址资源池中是否还有可用的地址。如果没有，则这次与 Internet 的通信就无法进行（根据前面的分析，这种可能性是非常非常小的）。如果有，则 NAT 会在公有地址资源池中选中一个公有 IP 地址，并在动态地址映射表中创建一个表项，该表项反映了该公有 IP 地址与该用户的私有 IP 地址之间的映射关系。当该用户结束了与 Internet 的通信后，NAT 必须将该表项从动态地址映射表中清除，同时将该表项中的公有 IP 地址释放回公有地址资源池。简而言之，使用动态 NAT 技术时，同一个公有 IP 地址可以分配给不同的私网用户使用，但在使用的时间上必须错开。

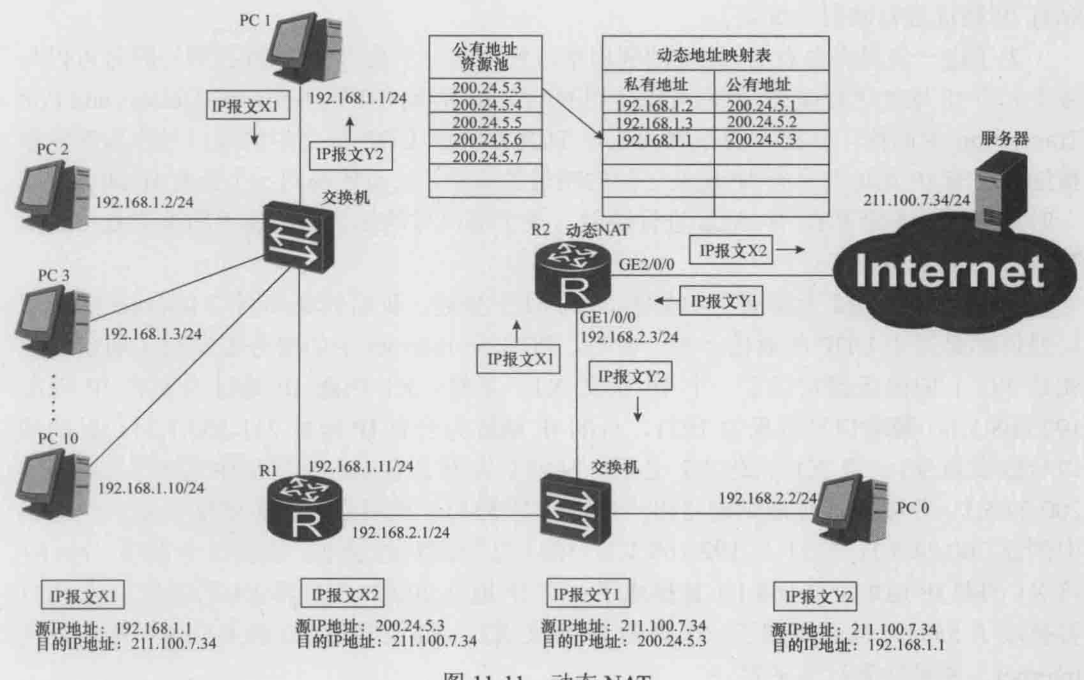


图 11-11　动态 NAT

图 11-11 中，PC 1 在某一时刻向 Internet 中的服务器发起了通信，也就是 PC 1 向服务器发送了一个 IP 报文 X1。显然，X1 的源 IP 地址为私有 IP 地址 192.168.1.1，目的 IP 地址为公有 IP 地址 211.100.7.34。当 X1 到达 R2 之后，NAT 在其公有地址资源池中选中了 IP 地址 200.24.5.3，然后在其动态地址映射表中创建了 200.24.5.3 与 192.168.1.1 之间的映射表项。根据这个表项，NAT 将 X1 的源 IP 地址 192.168.1.1 替换成了公有 IP 地址 200.24.5.3，从而得到了一个新的 IP 报文 X2。X2 会通过 R2 的 GE2/0/0 接口去往 Internet，并最终到达服务器。

当服务器向 PC 1 返回一个 IP 报文 Y1 时，Y1 的源 IP 地址应为公有 IP 地址 211.100.7.34，目的 IP 地址应为公有 IP 地址 200.24.5.3。Y1 进入 R2 后，NAT 会在动态地址映射表中查找 Y1 的目的 IP 地址 200.24.5.3，并发现其对应的私有 IP 地址为 192.168.1.1。然后，NAT 会将 Y1 的目的 IP 地址 200.24.5.3 替换为私有 IP 地址 192.168.1.1，从而得到一个新的 IP 报文 Y2。Y2 会通过 R2 的 GE1/0/0 接口去往私网，并最终到达 PC 1。注意，当 PC 1 完成了与服务器的通信后，NAT 必须清除其动态地址映射表中关于公有 IP 地址 200.24.5.3 的表项，并将公有 IP 地址 200.24.5.3 释放回公有地址资源池。

11.2.4　NAPT

如图 11-12 所示，公司进一步发展，用户数已经超过了 200，但可供公司使用的公有 IP 地址仍然只有原来那 7 个。由于用户数量太多，同一时刻需要与 Internet 进行通信的用户数几乎总是会超过 7 个。显然，在这种情况下，动态 NAT 技术也是无法满足需求

的。我们知道，无论是静态 NAT 还是动态 NAT，同一时刻一个公有 IP 地址只能与一个私有 IP 地址进行映射（绑定）。

为了进一步提高公有 IP 地址的利用率，使得同一个公有 IP 地址在同一时刻可以与多个私有 IP 地址进行映射，我们可以使用 NAPT 技术。NAPT 是 Network Address and Port Translation 的简称，其根本的原理就是将 TCP 报文或 UDP 报文中的端口号作为映射参数纳入公有 IP 地址与私有 IP 地址之间的映射关系中，从而使得同一个公有 IP 地址在同一时刻可以与多个私有 IP 地址进行映射。关于端口号的知识，请读者朋友们复习一下 7.2.5 小节的内容。

图 11-12 中，R2 上部署了 NAPT。为了便于描述，我们假定私网与 Internet 的应用层通信都是基于 UDP 的通信。某一时刻，PC 1 向 Internet 中的服务器发起了通信，也就是 PC 1 向服务器发送了一个 IP 报文 X1。显然，X1 的源 IP 地址为私有 IP 地址 192.168.1.1，源端口号假设为 1031，目的 IP 地址为公有 IP 地址 211.100.7.34，目的端口号假设为 Z1。当 X1 到达 R2 之后，NAPT 在其公有地址资源池中选中了 IP 地址 200.24.5.1，并根据某种规则确定出一个端口号 5531，然后在其动态地址及端口映射表中创建 200.24.5.1：5531 与 192.168.1.1：1031 之间的映射表项。根据这个表项，NAPT 将 X1 的源 IP 地址 192.168.1.1 替换成了公有 IP 地址 200.24.5.1，将 X1 的源端口号 1031 替换成了 5531，从而得到了一个新的 IP 报文 X2。X2 会通过 R2 的 GE2/0/0 接口去往 Internet，并最终到达服务器。

当服务器向 PC 1 返回一个 IP 报文 Y1 时，Y1 的源 IP 地址应为公有 IP 地址 211.100.7.34，源端口号假设为 Z2，目的 IP 地址应为公有 IP 地址 200.24.5.1，目的端口号应为 5531。Y1 进入 R2 后，NAPT 会在动态地址及端口映射表中查找 Y1 的目的 IP 地址 200.24.5.1 及目的端口号 5531，并发现它应该映射到 192.168.1.1：1031。于是，NAPT 会将 Y1 的目的 IP 地址 200.24.5.1 替换成私有 IP 地址 192.168.1.1，将目的端口号 5531 替换为 1031，从而得到一个新的 IP 报文 Y2。Y2 会通过 R2 的 GE1/0/0 接口去往私网，并最终到达 PC 1。

图 11-12 中，假设在 PC 1 与服务器的通信过程中，PC 2 也向 Internet 中的服务器发起了通信，也就是 PC 2 向服务器发送了一个 IP 报文 U1。显然，U1 的源 IP 地址为私有 IP 地址 192.168.1.2，源端口号假设为 1540，目的 IP 地址为公有 IP 地址 211.100.7.34，目的端口号假设为 Z3。当 U1 到达 R2 之后，NAPT 在其公有地址资源池中还是选中了 IP 地址 200.24.5.1，并根据某种规则确定出一个新的端口号 5532，然后在其动态地址及端口映射表中创建 200.24.5.1：5532 与 192.168.1.2：1540 之间的映射表项。根据这个表项，NAPT 将 U1 的源 IP 地址 192.168.1.2 替换成了公有 IP 地址 200.24.5.1，将 U1 的源端口号 1540 替换成了 5532，从而得到了一个新的 IP 报文 U2。U2 会通过 R2 的 GE2/0/0 接口去往 Internet，并最终到达服务器。

当服务器向 PC 2 返回一个 IP 报文 V1 时，V1 的源 IP 地址应为公有 IP 地址 211.100.7.34，源端口号假设为 Z4，目的 IP 地址应为公有 IP 地址 200.24.5.1，目的端口号应为 5532。V1 进入 R2 后，NAPT 会在动态地址及端口映射表中查找 V1 的目的 IP 地址 200.24.5.1 及目的端口号 5532，并发现它应该映射到 192.168.1.2：1540。于是，NAPT 会将 V1 的目的 IP 地址 200.24.5.1 替换成私有 IP 地址 192.168.1.2，将目的端口号 5532

替换成 1540，从而得到一个新的 IP 报文 V2。V2 会通过 R2 的 GE1/0/0 接口去往私网，并最终到达 PC 2。

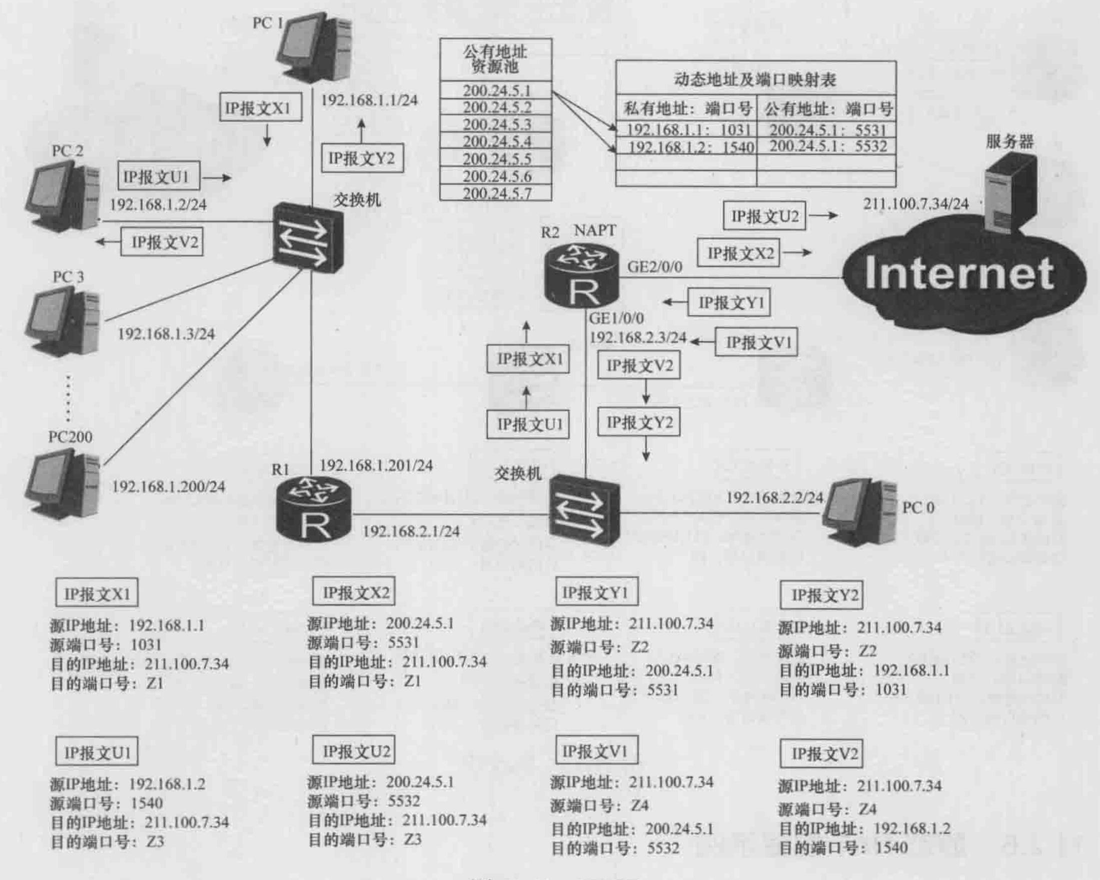


图 11-12　NAPT

11.2.5　Easy IP

Easy IP 技术是 NAPT 的一种简化情况。如图 11-13 所示，Easy IP 无需建立公有 IP 地址资源池，因为 Easy IP 只会用到一个公有 IP 地址，该 IP 地址就是路由器 R2 的 GE2/0/0 接口的 IP 地址。Easy IP 也会建立并维护一张动态地址及端口映射表，并且，Easy IP 会将这张表中的公有 IP 地址绑定成 R2 的 GE2/0/0 接口的 IP 地址。R2 的 GE2/0/0 接口的 IP 地址如果发生了变化，那么，这张表中的公有 IP 地址也会自动跟着变化。GE2/0/0 接口的 IP 地址可以是手工配置的，也可以是动态分配的。

其他方面，Easy IP 都是与 NAPT 完全一样的，这里不再赘述。图 11-13 中所示的 PC 1 和 PC 2 与公网服务器进行通信的例子，也与图 11-12 中所示的例子几乎完全一样，相信读者一看就能明白。

动态地址及端口映射表	
私有地址：端口号	公有地址：端口号
192.168.1.1：1031	200.24.5.1：5531
192.168.1.2：1540	200.24.5.1：5532

IP报文X1
源IP地址：192.168.1.1
源端口号：1031
目的IP地址：211.100.7.34
目的端口号：Z1

IP报文X2
源IP地址：200.24.5.1
源端口号：5531
目的IP地址：211.100.7.34
目的端口号：Z1

IP报文Y1
源IP地址：211.100.7.34
源端口号：Z2
目的IP地址：200.24.5.1
目的端口号：5531

IP报文Y2
源IP地址：211.100.7.34
源端口号：Z2
目的IP地址：192.168.1.1
目的端口号：1031

IP报文U1
源IP地址：192.168.1.2
源端口号：1540
目的IP地址：211.100.7.34
目的端口号：Z3

IP报文U2
源IP地址：200.24.5.1
源端口号：5532
目的IP地址：211.100.7.34
目的端口号：Z3

IP报文V1
源IP地址：211.100.7.34
源端口号：Z4
目的IP地址：200.24.5.1
目的端口号：5532

IP报文V2
源IP地址：211.100.7.34
源端口号：Z4
目的IP地址：192.168.1.2
目的端口号：1540

图 11-13　Easy IP

11.2.6　静态 NAT 配置示例

如图 11-14 所示，路由器 Router 的左侧是私网，右侧是公网。为了让 PC 能够与公网上的服务器进行通信，我们需要在 Router 上配置静态 NAT，将私有 IP 地址 192.168.0.2 与公有 IP 地址 202.10.1.3 绑定起来。



图 11-14　静态 NAT 配置示例

1. 配置思路

在 Router 的公网侧接口 GE2/0/0 下配置静态 NAT，将私有 IP 地址 192.168.0.2 与公有 IP 地址 202.10.1.3 绑定起来。

2. 配置步骤

在 Router 的公网侧接口 GE2/0/0 下执行命令 **nat static global** *global-address* **inside**

host-address，该命令的作用是将公网 IP 地址 *global-address* 与私网 IP 地址 *host-addres* 进行绑定。

#配置 Router。

```
[Router] interface gigabitethernet 2/0/0
[Router-GigabitEthernet2/0/0] nat static global 202.10.1.3 inside 192.168.0.2
[Router-GigabitEthernet2/0/0] quit
```

这样就完成了静态 NAT 的配置。命令 **display nat static** 可用来查看公有 IP 地址与私有 IP 地址的映射关系。

```
<Router> display nat static
Static Nat Information:
Interface    : GigabitEthernet2/0/0
    Global IP/Port          : 202.10.1.3/----
    Inside IP/Port          : 192.168.0.2/----
    Protocol : ----
    VPN instance-name       : ----
    Acl number              : ----
    Netmask   : 255.255.255.0
  Description : ----

Total :      1
```

从回显信息中我们可以看到，公有 IP 地址 202.10.1.3 已经和私有 IP 地址 192.168.0.2 建立了映射关系。

11.3　练习题

1.（多选）关于 DHCP 协议，以下说法中正确的是？（　）

A．DHCP 协议的前身是 BOOTP 协议

B．DHCP Server 每次给 DHCP Client 分配一个 IP 地址时，只是跟 DHCP Client 订立了一个关于这个 IP 地址的租约

C．决定 IP 地址租约期长短的是 DHCP Server，而不是 DHCP Client

D. DHCP 中继代理的作用是在 DHCP Client 与 DHCP Server 之间进行 DHCP 消息的中转

E．如果没有 DHCP 中继代理，则 DHCP Client 与 DHCP Server 之间是无法传递 DHCP 消息的

2.（单选）DHCP Client 首次从 DHCP Server 获取 IP 地址时需要经历的 4 个阶段依次是？（　）

A．发现阶段，请求阶段，提供阶段，确认阶段

B．发现阶段，请求阶段，确认阶段，提供阶段

C．发现阶段，提供阶段，请求阶段，确认阶段

D．发现阶段，确认阶段，请求阶段，提供阶段

3.（多选）假设 DHCP Client 已经从 DHCP Server 获取了 IP 地址。那么，DHCP Client 因重新启动而需要再次获取上一次得到的 IP 地址时，则只需要经历下面哪些阶段？（　）

A．发现阶段　　B．请求阶段　　C．确认阶段　　D．提供阶段

4．（多选）关于 DHCP 中继，以下说法中正确的是？（　）

A．DHCP Server 和 DHCP 中继代理位于同一个网段（二层广播域），但它们都没有与 DHCP Client 位于同一个网段。在这种情况下，HDCP 是不能正常工作的

B．DHCP Client 和 DHCP 中继代理位于同一个网段，但它们都没有与 DHCP Server 位于同一个网段。在这种情况下，DHCP 是可以正常工作的

C．DHCP 中继代理是 DHCP 的必要组成部分。没有 DHCP 中继代理，DHCP 就无法正常工作

5．（多选）关于 NAT 技术，以下说法中正确的是？（　）

A．使用 NAT 技术时的基本场景是：公网内部进行通信

B．使用 NAT 技术时的基本场景是：私网内部进行通信

C．使用 NAT 技术时的基本场景是：私网与公网进行通信

D．NAT 技术可以在一定程度上节约公有 IP 地址资源

E．IP 地址空间中，私有 IP 地址的数量远小于公有 IP 地址的数量。NAT 技术的使用，可以在很大程度上节约私有 IP 地址资源

F．Easy IP 是 NAPT 的一种特殊情况

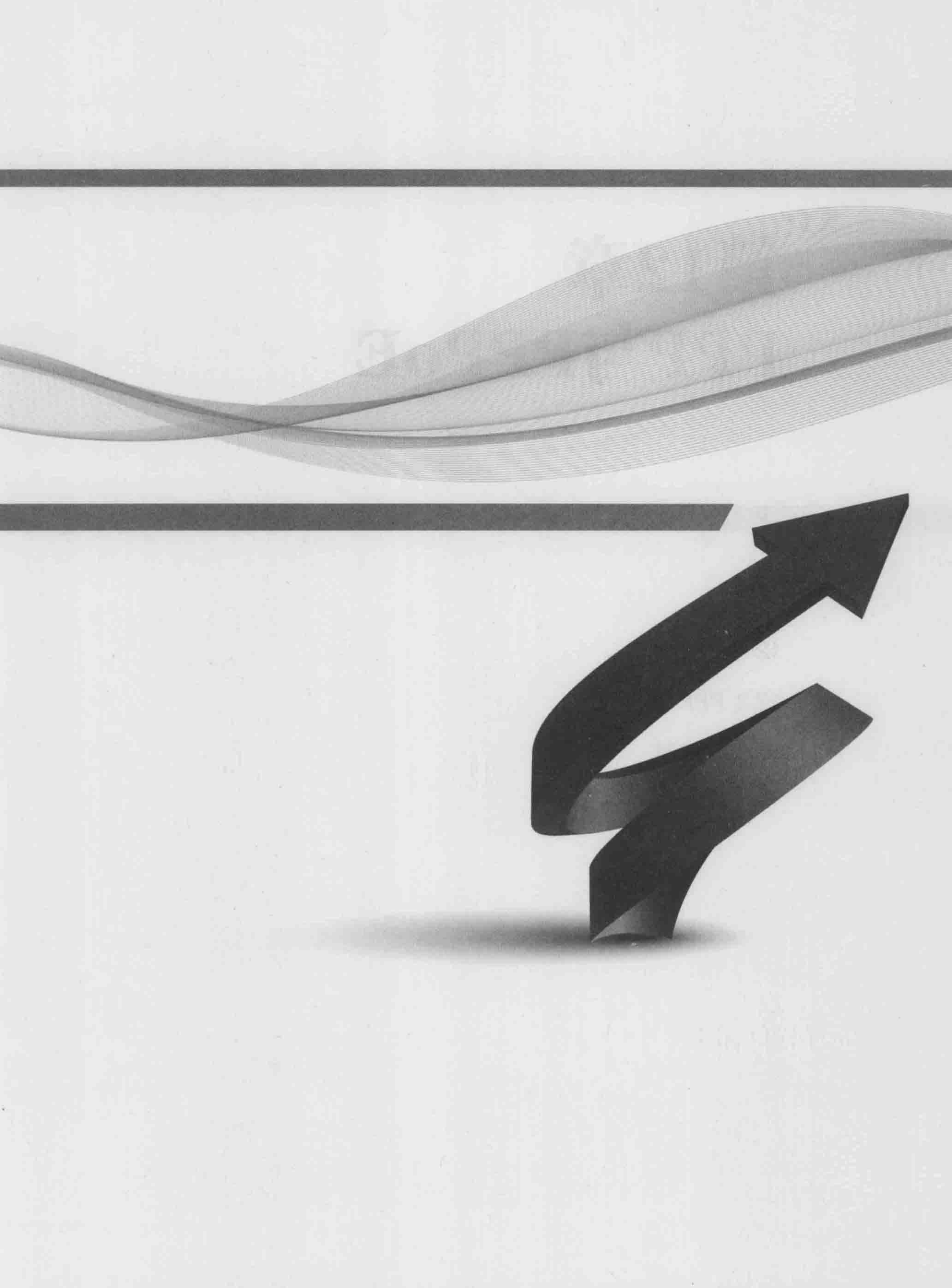
6．（单选）以下选项中，哪一项暗含了公有 IP 地址利用率从低到高的顺序？（　）

A．动态 NAT，静态 NAT，NAPT

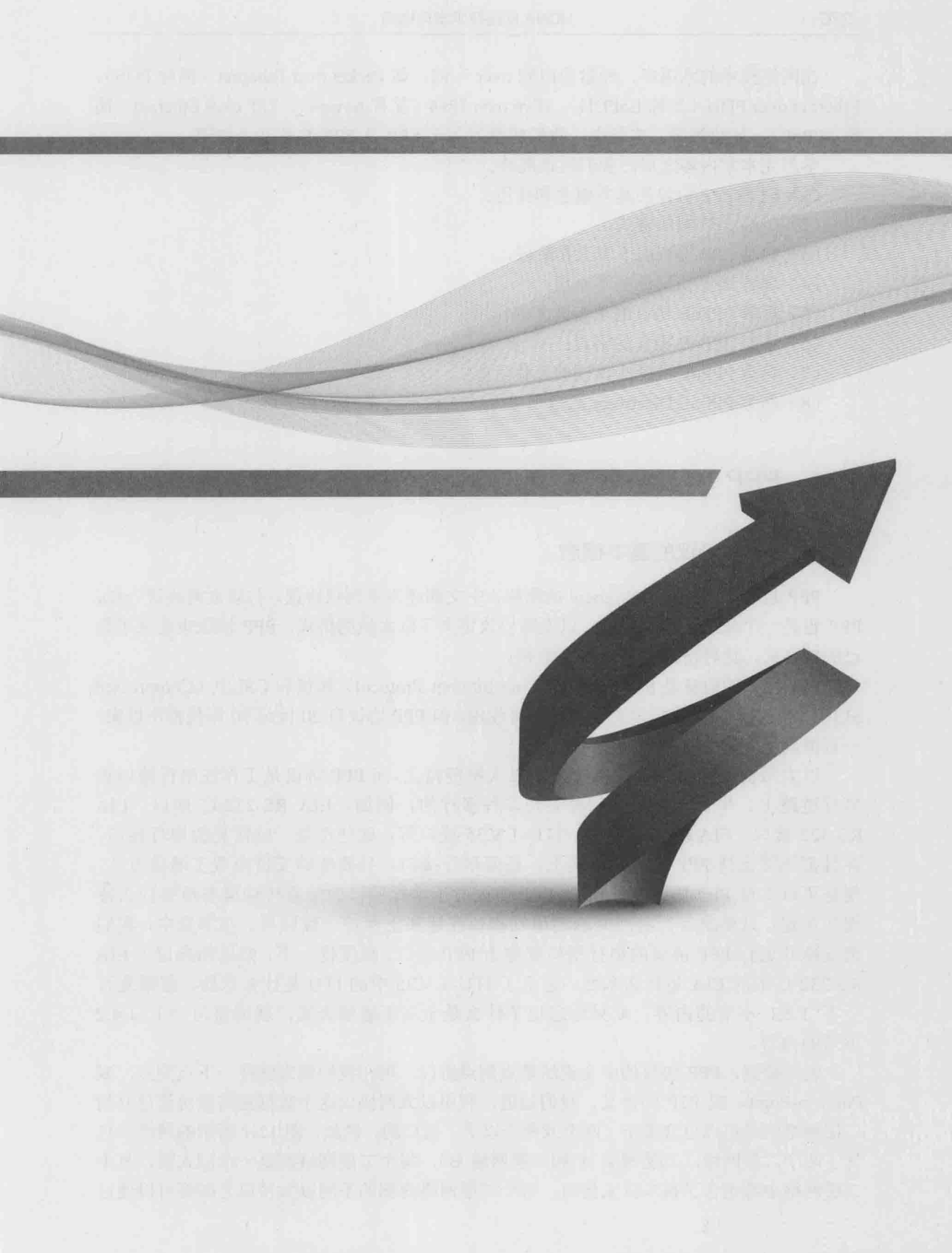
B．NAPT，动态 NAT，静态 NAT

C．静态 NAT，NAPT，动态 NAT

D．静态 NAT，动态 NAT，NAPT



第12章 PPP与PPPoE



在网络技术的术语中，经常会碰到 over 一词，如 Packet over Ethernet（简称 POS）、Ethernet over PDH（简称 EoPDH）、IPv6 over IPv4（简称 6over4）、PPP over Ethernet（简称 PPPoE），如此等等。本章中，我们将学习关于 PPP 及 PPPoE 的基本知识。

学习完本章内容之后，我们应该能够：

（1）理解 PPP 协议的基本概念和作用；

（2）了解 PPP 帧的格式；

（3）熟悉 PPP 协议的 5 个工作阶段；

（4）理解 PAP 认证的基本原理；

（5）理解 PPPoE 协议的基本概念的作用；

（6）了解 PPPoE 报文的格式；

（7）熟悉 PPPoE 的两个不同的工作阶段；

（8）熟悉 PPPoE Discovery 阶段所涉及的 4 种不同类型的 PPPoE 报文。

12.1 PPP

12.1.1 PPP 协议的基本概念

PPP 是 Point-to-Point Protocol 的简称，中文翻译为点到点协议。与以太网协议一样，PPP 也是一个数据链路层协议。以太网协议定义了以太帧的格式，PPP 协议也定义了自己的帧格式，这种格式的帧称为 PPP 帧。

PPP 协议的前身是 SLIP（Serial Line Internet Protocol）协议和 CSLIP（Compressed SLIP）协议，前两种协议现在已基本不再使用，但 PPP 协议自 20 世纪 90 年代推出以来，一直得到了广泛的应用。

以太网协议工作在以太网接口和以太网链路上，而 PPP 协议是工作在串行接口和串行链路上。串行接口本身的种类是多种多样的，例如，EIA RS-232-C 接口、EIA RS-422 接口、EIA RS-423 接口、ITU-T V.35 接口等，这些都是一些常见的串行接口，并且都能够支持 PPP 协议。事实上，任何串行接口，只要能够支持全双工通信方式，便是可以支持 PPP 协议的。另外，PPP 协议对于串行接口的信息传输速率没有什么特别的规定，只要求串行链路两端的串行接口在速率上保持一致即可。在本章中，我们把支持并运行 PPP 协议的串行接口统称为 PPP 接口。顺便提一下，如果你忘记了 EIA RS-232-C 中的 EIA 是什么东西，忘记了 ITU-T V.35 中的 ITU 是什么东西，就请复习一下 1.2.1 小节的内容；如果你忘记了什么是全双工通信方式，就请复习一下 1.4.2 小节的内容。

刚才提到，PPP 协议的中文说法是点到点协议，现在我们就来解释一下点到点，或 Point-to-Point，或 P2P 的含义。我们知道，利用以太网协议这个数据链路层协议建立的二层网络中是可以包含多个（两个或两个以上）接口的。例如，图 12-1 所示的网络中包含了两个二层网络（二层网络 A 和二层网络 B），每个二层网络都是一个以太网，每个二层网络中都包含了很多以太接口，每个二层网络内部的不同以太接口之间都可以通过

交互以太帧的方式来实现二层通信。因此，我们也把以太网称为一种多点接入网络（Multi-Access Network），其含义是指这样的网络中可以包含多个（两个或两个以上）接口，且网络内部的任意两个接口之间都可以进行二层通信。

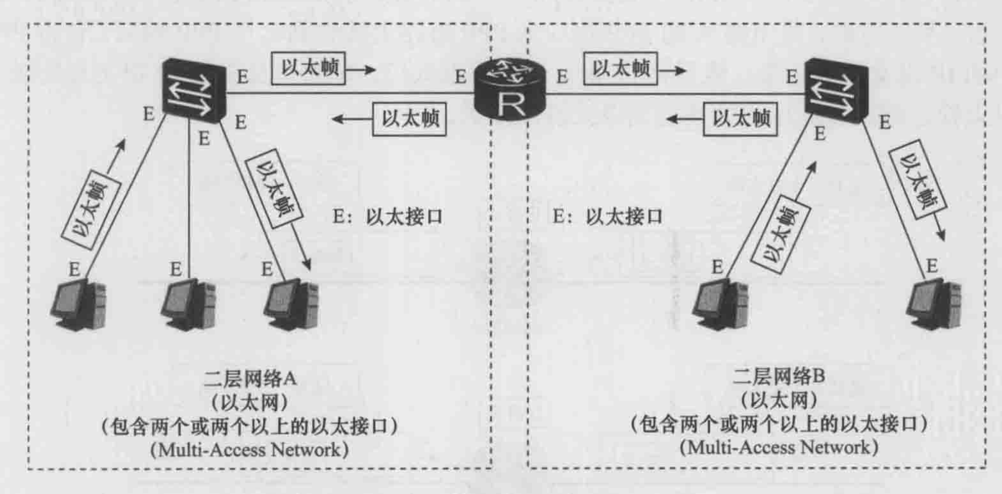


图 12-1　以太网是一种 Multi-Access Network

利用 PPP 协议建立的二层网络称为 PPP 网络。一个 PPP 网络包含且只能包含两个 PPP 接口，连接这两个接口的链路称为 PPP 链路，这两个接口通过交互 PPP 帧来实现二层通信。例如，图 12-2 所示的网络中包含了 3 个二层网络，二层网络 A 是一个以太网，二层网络 B 是一个 PPP 网络，二层网络 C 是另外一个 PPP 网络。由于一个 PPP 网络包含且只能包含两个 PPP 接口，每个接口被简化地称为一个“点”，所以一个 PPP 网络也经常被称为一个“点到点网络”或“P2P 网络”。

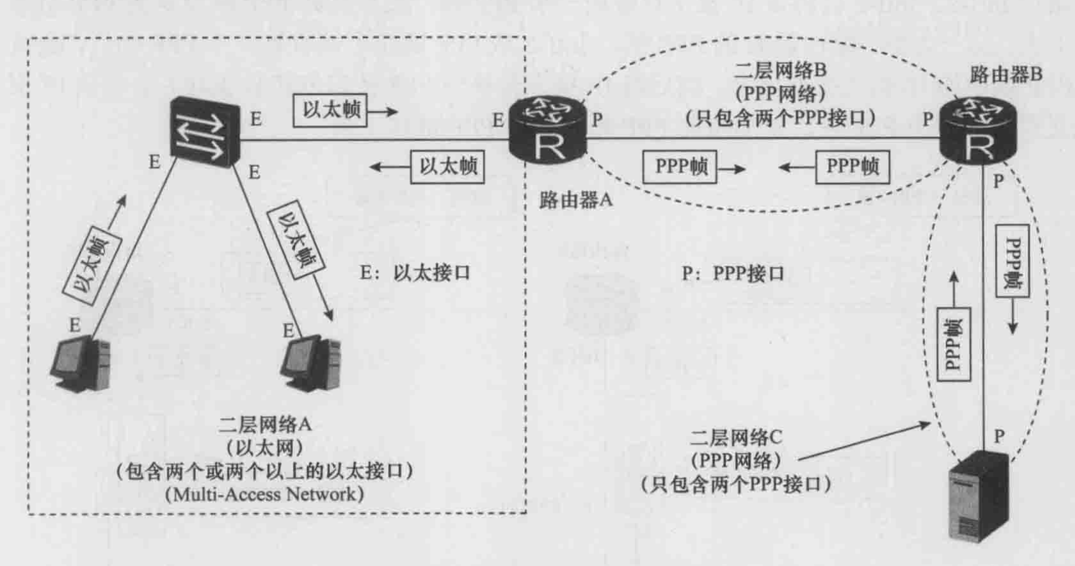


图 12-2　Multi-Access 网络与 P2P 网络

在图 12-2 中，路由器 A 有两个接口，一个是以太口，另一个是 PPP 接口。图 12-3 显示了图 12-2 中路由器 A 的基本工作过程。如图 12-3 所示，路由器 A 的以太口从以太链路上接收到一个以太帧后，会将以太帧中的 IP 报文提取出来，然后将 IP 报文转移至 PPP 接口。PPP 接口会将该 IP 报文封装成一个 PPP 帧，然后将此 PPP 帧发送到 PPP 链路上去。另一方面，路由器 A 的 PPP 接口从 PPP 链路上接收到一个 PPP 帧后，会将 PPP 帧中的 IP 报文提取出来，然后将 IP 报文转移至以太口。以太口会将该 IP 报文封装成一个以太帧，然后将此以太帧发送到以太链路上去。

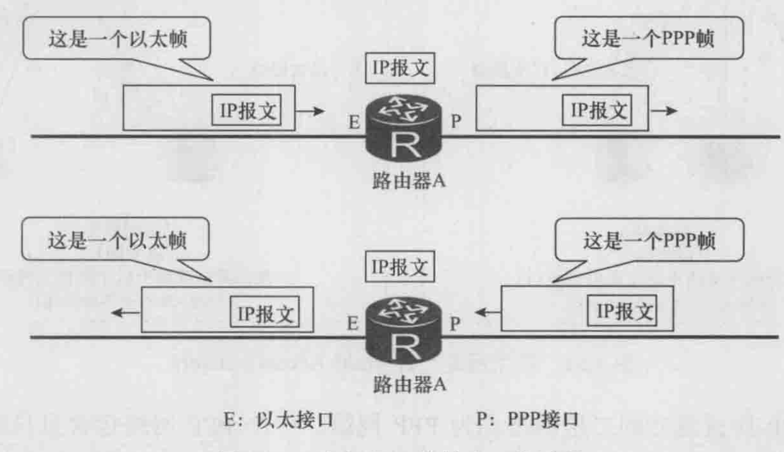


图 12-3 路由器 A 的基本工作过程

图 12-2 中，路由器 B 有两个接口，这两个接口都是 PPP 接口。图 12-4 显示了图 12-2 中路由器 B 的基本工作过程。如图 12-4 所示，路由器 B 的 PPP 接口 Intf-1 从 PPP 链路上接收到一个 PPP 帧后，会将 PPP 帧中的 IP 报文提取出来，然后将 IP 报文转移至 PPP 接口 Intf-2。Intf-2 会将该 IP 报文封装成一个 PPP 帧，然后将此 PPP 帧发送到 PPP 链路上去。另一方面，路由器 B 的 PPP 接口 Intf-2 从 PPP 链路上接收到一个 PPP 帧后，会将 PPP 帧中的 IP 报文提取出来，然后将 IP 报文转移至 PPP 接口 Intf-1。Intf-1 会将该 IP 报文封装成一个 PPP 帧，然后将此 PPP 帧发送到 PPP 链路上去。

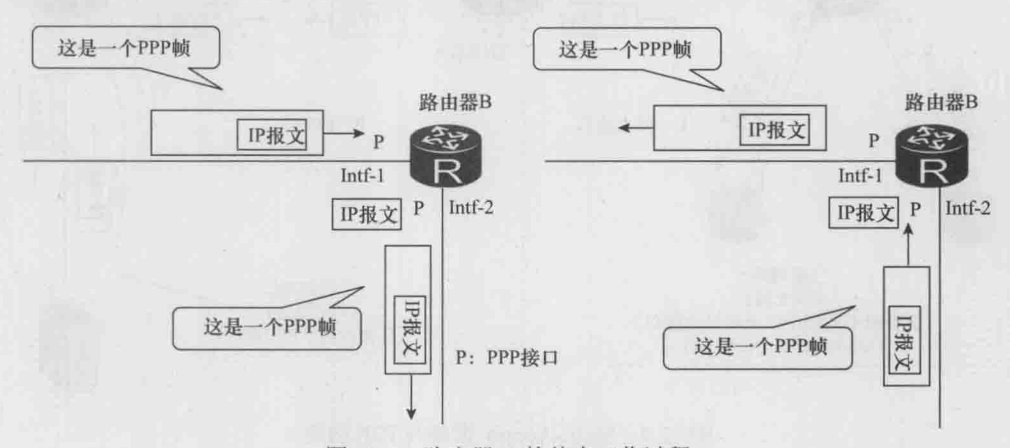


图 12-4 路由器 B 的基本工作过程

从上面的描述我们可以看到，PPP接口是数据链路层（二层）通信的终结点，所以我们也说，PPP接口是三层接口。认识到这一点是非常重要的。

需要说明的是，PPP协议还包含了若干个附属协议，这些附属协议也称为成员协议。PPP协议的成员协议主要包括一个被称为LCP（Link Control Protocol）的链路控制协议，以及一系列的被称为NCP（Network Control Protocol）的网络控制协议。

另外需要说明的是，PPP协议对于PPP链路的长度是没有规定的。PPP链路经常应用在广域网连接中；PPP技术被称为是一种广域网技术。

12.1.2　PPP帧的格式

图12-5显示了PPP帧的格式，下面是关于PPP帧格式中各个字段的含义的描述。

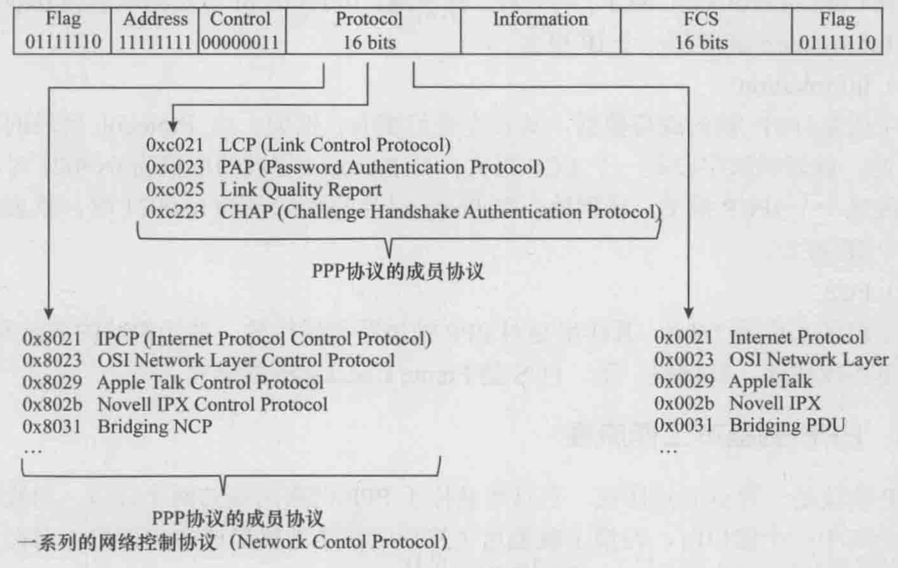


图12-5　PPP帧的格式

（1）Flag

该字段的长度为8bit，取值固定为0x7e。该字段标志了一个PPP帧的开始或结束；它既标志了当前PPP帧的开始，同时也标志了前一个PPP帧的结束。

注意，一个PPP帧的Information字段中是不允许出现0x7e的，因为这样就会使得PPP帧的接收方错误地把这个0x7e当成Flag来对待。那么，如果一个PPP帧的Information字段中需要包含0x7e时，那该怎么呢？遇到这样的情况时，Information字段中的0x7e就必须经过“转意”处理。关于该如何进行“转意”处理，我们这里不作描述。总之，经过“转意”处理之后，Information字段中就不再可能出现0x7e了。

（2）Address

该字段的长度为8bit，取值固定为0xff。需要注意的是，该字段并非是一个MAC地址，但它具有广播地址的含义，意思是“所有的接口”。

PPP帧是在一条单一的PPP链路上固定地从此接口运动到彼接口，因此PPP帧不像以太帧那样包含了源MAC地址和目的MAC地址这些信息。事实上，PPP接口根本就不

需要属于自己的MAC地址，MAC地址对于PPP接口来说毫无意义。

（3）Control

该字段的长度为8bit，取值固定为0x03。该字段并没有什么特别的作用，至于其取值为何固定为0x03，我们这里不做解释。

（4）Protocol

该字段的长度为16bit，它的取值决定了Information字段包含的是什么样的协议报文。该字段的作用类似于以太帧中的类型字段。

图12-5中举例显示了Protocol字段的不同取值所对应的不同协议。例如，当Protocol字段的取值为0xc021时，就表明Information字段是一个LCP报文；当Protocol字段的取值为0x8021时，就表明Information字段是一个IPCP报文。IPCP是网络控制协议（Network Control Protocol，NCP）的一种。特别地，当Protocol字段的取值为0x0021时，就表明Information字段是一个IP报文。

（5）Information

该字段是PPP帧的载荷数据，其长度是可变的。例如，当Protocol字段的取值为0xc021时，就表明该字段是一个LCP报文；当Protocol字段的取值为0x8021时，就表明该字段是一个IPCP报文。特别地，当Protocol字段的取值为0x0021时，就表明该字段是一个IP报文。

（6）FCS

该字段的长度为16bit，其作用是对PPP帧进行差错校验。关于校验的方法和过程，我们这里不做描述。顺便提一下，FCS是Frame Checksum的缩写。

12.1.3 PPP的基本工作流程

PPP协议是一种点到点协议，它只涉及位于PPP链路两端的两个接口。当我们在分析和讨论其中一个接口时，习惯上就把这个接口叫做本地接口或本端接口，而把另一个接口叫做对端接口或远端接口，或简称为Peer。

通过串行链路连接起来的本地接口和对端接口在上电之后，并不能马上就开始相互发送携带有诸如IP报文这样的网络层数据单元的PPP帧。本地接口和对端接口在开始相互发送携带有诸如IP报文这样的网络层数据单元的PPP帧之前，必须经过一系列复杂的协商过程（甚至还可能包括认证过程），这一过程也称为PPP的基本工作流程，如图12-6所示。

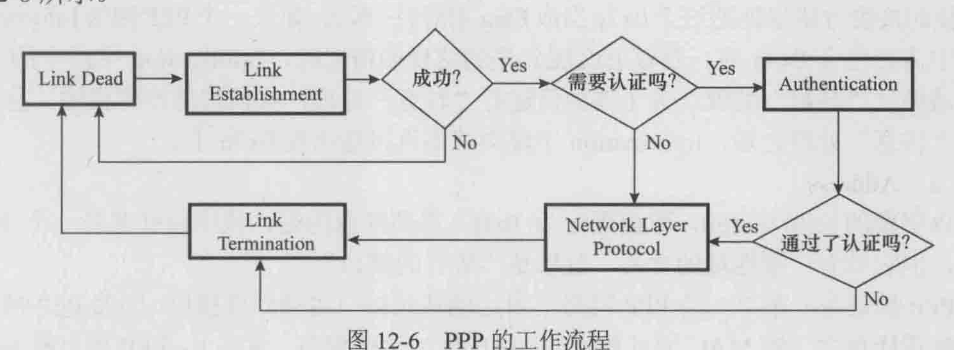


图12-6 PPP的工作流程

从图 12-6 中我们可以看到，PPP 基本工作流程总共包含了 5 阶段，分别是：Link Dead 阶段（即链路关闭阶段），Link Establishment 阶段（即链路建立阶段），Authentication 阶段（即认证阶段），Network Layer Protocol 阶段（即网络层协议阶段），Link Termination 阶段（即链路终结阶段）。

PPP 基本工作流程的第一个阶段是 Link Dead 阶段。在此阶段，PPP 接口的物理层功能尚未进入正常状态。只有当本端接口和对端接口的物理层功能都进入正常状态之后，PPP 才能进入到下一个工作阶段，即 Link Establishment 阶段。

当本端接口和对端接口的物理层功能都进入正常状态之后，PPP 便会自动进入到 Link Establishment 阶段。在此阶段，本端接口会与对端接口相互发送携带有 LCP 报文的 PPP 帧。简单地说，此阶段也就是双方交互 LCP 报文的阶段。通过 LCP 报文的交互，本端接口会与对端接口协商若干基本而重要的参数，以确保 PPP 链路可以正常工作。例如，本端接口会与对端接口对 MRU（Maximum Receive Unit）这个参数进行协商。所谓 MRU，就是 PPP 帧中 Information 字段所允许的最大长度（字节数）。如果本端接口因为某种原因而要求所接收到的 PPP 帧的 Information 字段的长度不得超过 1 800 字节（即本端接口的 MRU 为 1 800），而对端接口却发送了 Information 字段为 2 000 字节的 PPP 帧，那么，在这种情况下，本端接口就无法正确地接收和处理这个 Information 字段为 2 000 字节的 PPP 帧，通信就会因此而产生故障。因此，为了避免这种情况的发生，本端接口和对端接口在 Link Establishment 阶段就必须对 MRU 这个参数进行协商并取得一致意见，此后，本端接口就不会发送 Information 字段超过对端 MRU 的 PPP 帧，对端接口也不会发送 Information 字段超过本端 MRU 的 PPP 帧。

在 Link Establishment 阶段，本端接口和对端接口还必须约定好是直接进入到 Network Layer Protocol 阶段呢，还是先进入 Authentication 阶段，再进入到 Network Layer Protocol 阶段。如果需要进入到 Authentication 阶段，还必须约定好使用什么样的认证协议来进行认证。

如果 PPP 的 Link Establishment 阶段顺利地结束了，并且 PPP 协议的双方约定无需进行认证，或者双方顺利地结束了认证阶段，那么 PPP 就会自动进入到 Network Layer Protocol 阶段。在 Network Layer Protocol 阶段，PPP 协议的双方会首先通过 NCP（Network Control Protocol）协议来对网络层协议的参数进行协商，协商一致之后，双方才能够在 PPP 链路上传递携带有相应的网络层协议数据单元的 PPP 帧。

有很多种情况都会导致 PPP 进入到 Link Termination 阶段，例如认证阶段未能顺利完成，例如链路的信号质量太差，例如网络管理员需要主动关闭链路，如此等等。

在下面的 3 个小节中，我们将分别对 PPP 的 Link Establishment 阶段、Authentication 阶段和 Network Layer Protocol 阶段进行分析和描述。

12.1.4 PPP 之链路建立阶段

LCP 协议是 PPP 协议的主要成员协议之一。在 PPP 的 Link Establishment 阶段，本端接口和对端接口总是发送携带有 LCP 报文的 PPP 帧。LCP 报文的格式如图 12-7 所示。

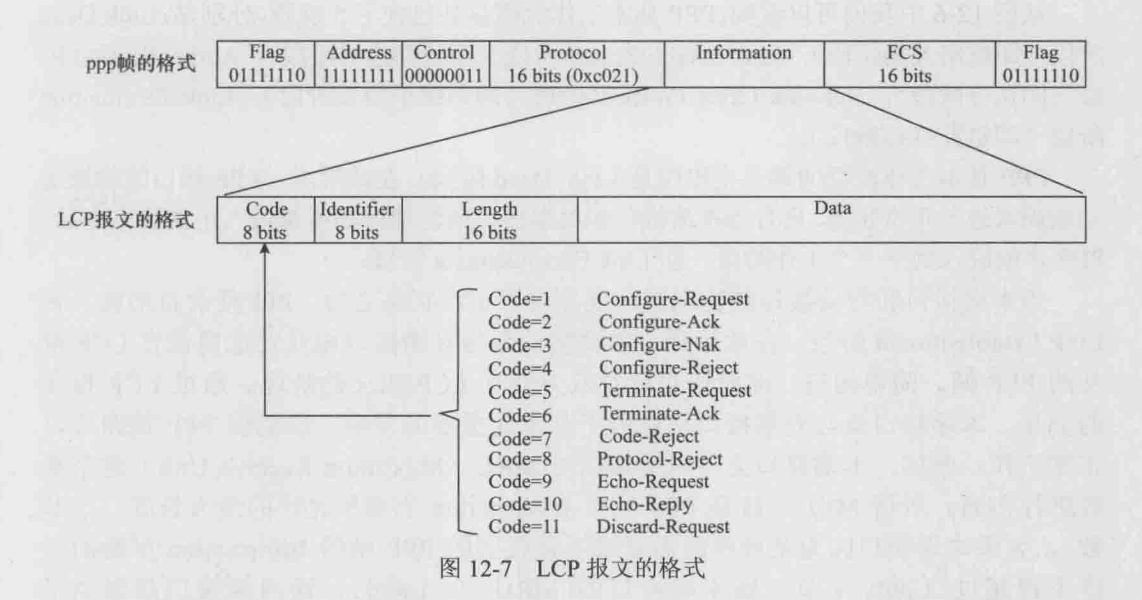


图 12-7 LCP 报文的格式

从图 12-7 中我们可以看到，如果 PPP 帧的 Protocol 字段的值为 0xc021，则表明 PPP 帧的 Information 字段的内容是一个 LCP 报文。LCP 报文包含了 4 个字段，分别是 Code 字段、Identifier 字段、Length 字段和 Data 字段。LCP 报文中，Code 字段可以取不同的值，用以区分不同类型的 LCP 报文。例如，如果 Code 字段的取值为 1，则说明这是一个 Configure-Request 报文；如果 Code 字段的取值为 2，则说明这是一个 Configure-Ack 报文；如果 Code 字段的取值为 3，则说明这是一个 Configure-Nak 报文，如此等等。从图 12-7 中我们可以看到，LCP 报文的类型一共有 11 种。

LCP 报文中的 Identifier 字段是用来对本端接口发送的 LCP 报文和对端接口发送的回应报文进行匹配的。LCP 报文中的 Length 字段表明了该 LCP 报文的总长度（即 Code 字段、Identifier 字段、Length 字段和 Data 字段的长度之和）。LCP 报文中的 Data 字段的内容和长度会因 LCP 报文类型的不同而不同。

在 11 种 LCP 报文中，我们将抽取其中的 Configure-Request 报文（配置请求报文）进行分析和描述，因为这种报文在 Link Establishment 阶段扮演着主角的角色。如图 12-8 所示，在 Link Establishment 阶段，本端接口和对端接口都必须至少向对方发送一个 Configure-Request 报文，该报文中包含了发送方对于所有的配置参数的期望值。如果对方对于自己发送的 Configure-Request 报文回应了一个 Configure-Ack 报文，则说明对方已经认可了自己对于所有的配置参数的期望值；如果对方对于自己发送的 Configure-Request 报文回应了一个 Configure-Nak 报文，则说明对方否定了自己对于某些配置参数或所有的配置参数的期望值，这也意味着自己必须修改自己对于相应的配置参数的期望值，然后重新向对方发送一个 Configure-Request 报文，且等待对方的新的回应。此过程可以重复进行。如果最终双方都接收到了对方发送的 Configure-Ack 报文，则说明双方对于配置参数的协商已经取得一致，这同时也标志着 Link Establishment 阶段的顺利结束。

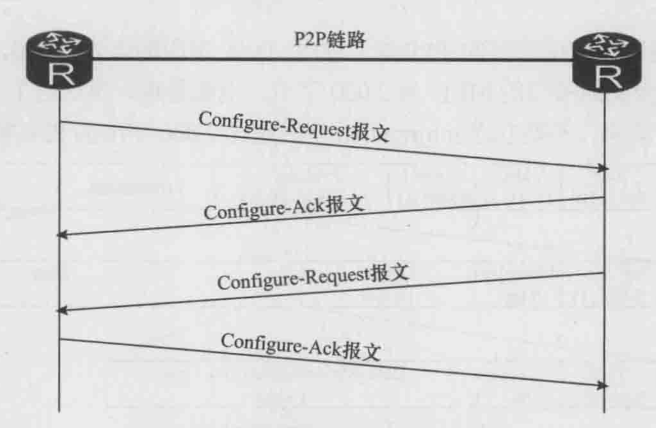


图 12-8　最简形式的链路建立过程

图 12-9 显示了 Configure-Request 报文的格式。注意，Configure-Request 报文是 11 种 LCP 报文中的一种，Code 字段的值为 1。Configure-Request 报文的 Data 字段总共可以包含最多 8 个配置选项，每一个配置选项其实就是一个需要协商的配置参数。每个配置选项都由 3 个字段组成，分别是 Type 字段、Length 字段和 Data 字段。如图 12-9 所示，如果 Type 字段的值为 1，则表明该配置选项是关于 MRU 这个参数的；如果 Type 字段的值为 2，则表明该配置选项是关于 Asynchronisation Control Character Map 这个参数的；如果 Type 字段的值为 3，则表明该配置选项是关于 Authentication Protocol 这个参数的……如果 Type 字段的值为 8，则表明该配置选项是关于 Address and Control Field Compression 这个参数的。

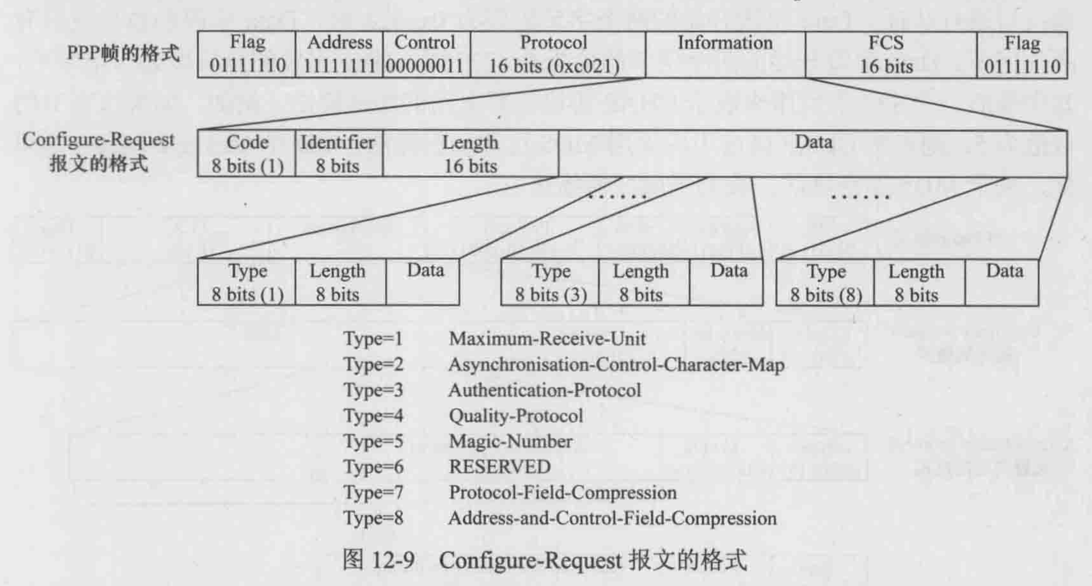


图 12-9　Configure-Request 报文的格式

我们来看看 MRU 这个配置选项。如图 12-10 所示，MRU 配置选项的 Type 字段的值为 1，Length 字段的值为 4（表明 MRU 配置选项的总长度为 4 个字节，其中 Type 字段占了 1 个字节，Length 字段本身占了 1 个字节，Data 字段占了 2 个字节），Data 字段的值就 MRU 这个参数。例如，如果 Data 字段的值为 1 800，就表明发送这个 Configure-Request 报文的接口的 MRU 为 1 800 字节，也就是说，发送这个 Configure-Request 报文的接口希望对端接口不要发送

Information 字段超过 1 800 字节的 PPP 帧；如果 Data 字段的值为 2 000，就表明发送这个 Configure-Request 报文的接口的 MRU 为 2 000 字节，也就是说，发送这个 Configure-Request 报文的接口希望对端接口不要发送 Information 字段超过 2 000 字节的 PPP 帧。

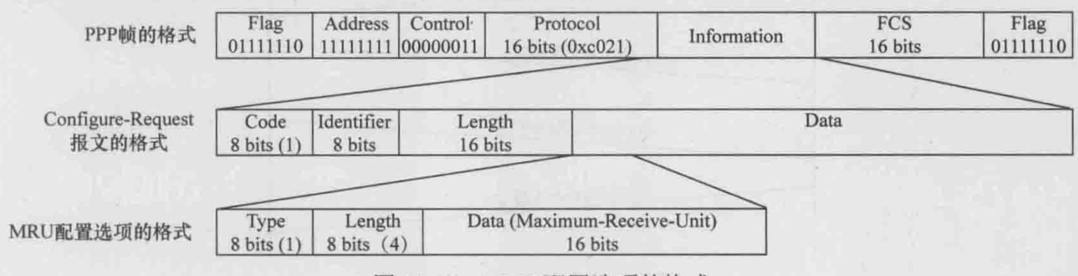


图 12-10 MRU 配置选项的格式

我们再来看看 Authentication Protocol 这个配置选项。如图 12-11 所示，Authentication Protocol 配置选项的 Type 字段的值为 3，Length 字段的值为可能为 4，可能为 5。如果 Data 字段开始的两个字节的值为 0xc023，则 Length 字段的值为 4；如果 Data 字段开始的两个字节的值为 0xc223，则 Length 字段的值为 5。

另一方面，如果 Data 字段开始的两个字节的值为 0xc023，则表明发送这个 Configure-Request 报文的接口希望在 PPP 的 Authentication 阶段采用 PAP 协议来对对端接口进行认证；如果 Data 字段开始的两个字节的值为 0xc223，则表明发送这个 Configure-Request 报文的接口希望在 PPP 的 Authentication 阶段采用 CHAP 协议来对对端接口进行认证。Data 字段开始的两个字节的值为 0xc023 时，Data 字段的总长度只有两个字节。Data 字段开始的两个字节的值为 0xc223 时，Data 字段的总长度为 3 个字节，其中最后一个字节的值用来表示 CHAP 协议需要采用的加密算法（例如，如果该字节的取值为 5，则表明 CHAP 协议中应采用 MD5 这种加密算法。MD 是 Message Digest 的简称，关于 MD5 加密算法，我们不做任何描述）。

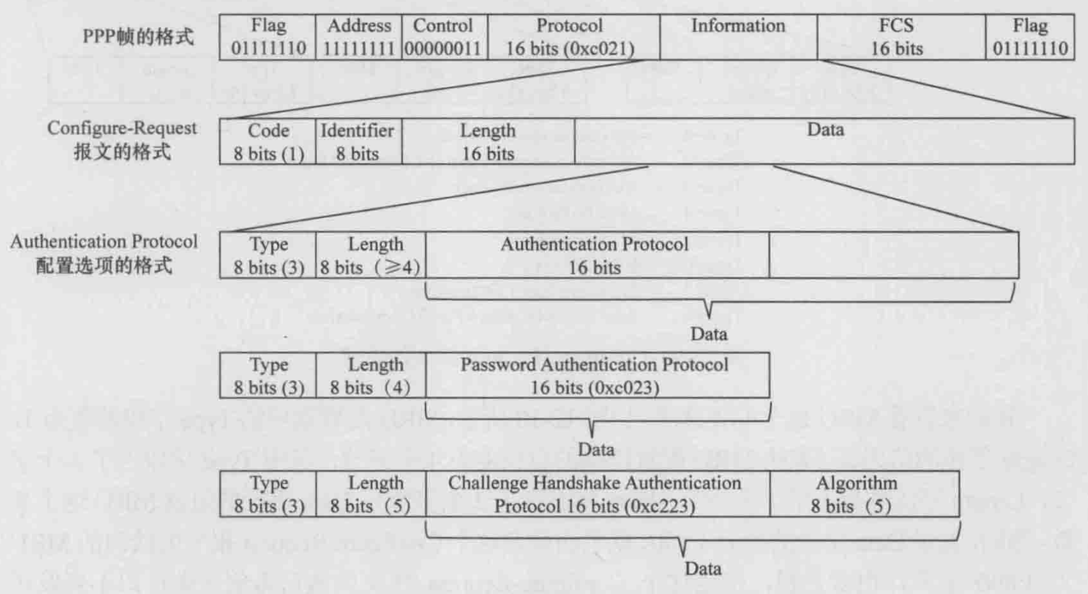


图 12-11 Authentication Protocol 配置选项的格式

需要说明的是，PPP 接口关于 Authentication Protocol 这个配置选项进行协商后的结果不一定是对称的。协商的结果可以是，此接口会对彼接口进行认证，但彼接口不会对此接口进行认证；也可以是，此接口不会对彼接口进行认证，但彼接口会对此接口进行认证；也可以是，此接口会对彼接口进行认证，同时彼接口也会对此接口进行认证；也可以是，此接口不会对彼接口进行认证，同时彼接口也不会对此接口进行认证。另外，此接口对彼接口进行认证时所采用的认证协议与彼接口对此接口进行认证时所采用的认证协议可以是同一种协议，也可以是不同的协议（例如，此接口是采用 PAP 协议对彼接口进行认证，但彼接口是采用 CHAP 协议对此接口进行认证）。

另外，需要特别说明的是，在 PPP 的 Link Establishment 阶段，所有需要协商的配置选项总共有 8 个（注：实质上只有 7 个，因为其中有一个是预留项），这 8 个选项是包含在同一个 Configure-Request 报文中一次性进行协商的，如图 12-9 所示。如果某些配置选项没有出现在 Configure-Request 报文中，则表明这些配置选项是取 PPP 协议规定的“缺省值”。例如，如果本端接口发送的 Configure-Request 报文中没有包含 MRU 配置选项，则表明本端接口的 MRU 是 1 500 字节（注：PPP 协议规定 MRU 的缺省值为 1 500 字节）。又例如，如果本端接口发送的 Configure-Request 报文中没有包含 Authentication Protocol 这个配置选项，则表明本端接口将不会对对端接口进行认证。

12.1.5 PPP 之认证阶段

如果 PPP 的 Link Establishment 阶段顺利结束了，并且某一接口要求对对端接口进行认证，或者双方都要求对对端接口进行认证，那么 PPP 就会进入到 Authentication 阶段（即认证阶段）。

Authentication 阶段涉及 PAP 和 CHAP 这两个认证协议，它们都是 PPP 协议的成员协议。我们这里仅以 PAP 协议为例来对 PPP 的 Authentication 阶段进行一个简单的描述。

我们先来看看 PAP 报文的格式，如图 12-12 所示。从图 12-12 中我们可以看到，PPP 帧的 Protocol 字段的值为 0xc023 时，PPP 帧的 Information 字段便是一个 PAP 报文。PAP 报文的 Code 字段是用来区分 PAP 报文的类型的。如果 Code 字段的值为 1，则表明 PAP 报文是一个 Authenticate-Request 报文；如果 Code 字段的值为 2，则表明 PAP 报文是一个 Authenticate-Ack 报文；如果 Code 字段的值为 3，则表明 PAP 报文是一个 Authenticate-Nak 报文。

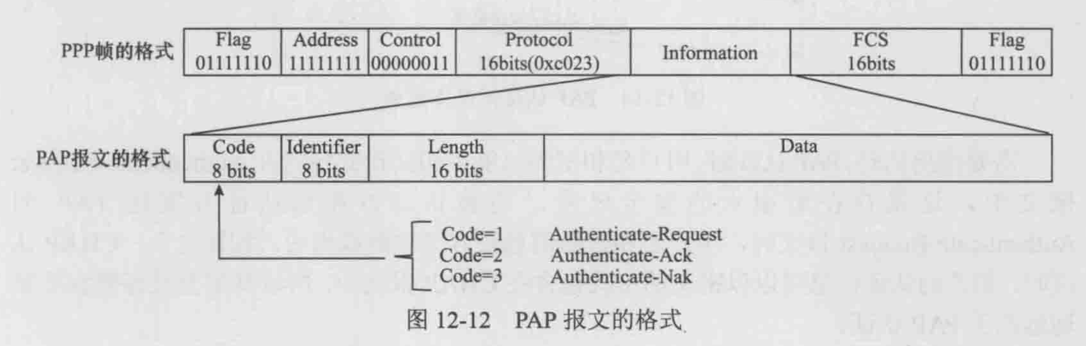


图 12-12 PAP 报文的格式

我们再来看看 Authenticate-Request 报文的格式，如图 12-13 所示。从图 12-13 中我

们可以看到，在 Authenticate-Request 报文中有这样的两个字段，一个是 Peer-ID 字段，另一个是 Password 字段。Peer-ID 字段的内容其实就是用户名，Password 字段的内容其实就是与用户名相对应的密码。关于 Authenticate-Ack 报文和 Authenticate-Nak 报文的格式，我们这里不做描述。

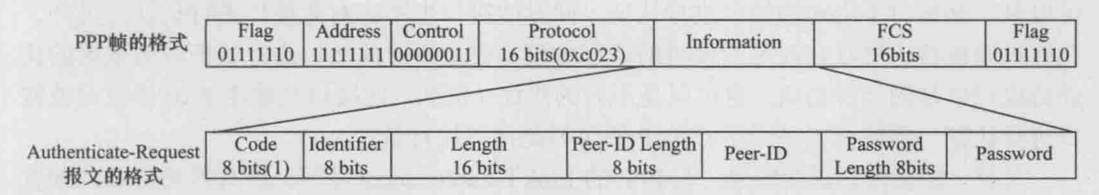


图 12-13 Authenticate-Request 报文的格式

如图 12-14 所示，如果在 PPP 的 Link Establishment 阶段，B 接口向 A 接口发送的 Configure-Request 报文中表明了 B 将对 A 进行 PAP 认证，并且 B 接口收到了 A 接口回应的 Configure-Ack 报文，那么在接下来的 Authentication 阶段，作为认证方的 B 就会对作为被认证方的 A 进行 PAP 认证。认证开始时，A 会向 B 发送包含了用户名和密码的 Authenticate-Request 报文。B 在接收到来自 A 的 Authenticate-Request 报文后，会从 Authenticate-Request 报文中提取出 A 提供的用户名和密码，并在自己的用户列表中查找 A 提供的用户名。如果 B 在自己的用户列表中不能查找到 A 提供的用户名，则 B 会向 A 回应一个 Authenticate-Nak 报文，这就意味着认证失败了。如果 B 在自己的用户列表中能够查找到 A 提供的用户名，但用户列表中相应用户的密码与 A 提供的密码不一致，那么 B 还是会向 A 回应一个 Authenticate-Nak 报文，表明认证失败。只有当 A 提供的用户名和密码完全匹配了 B 的用户列表中的相应条目时，B 才会向 A 回应一个 Authenticate-Ack 报文，这也标志着 A 成功地通过了 B 的认证。

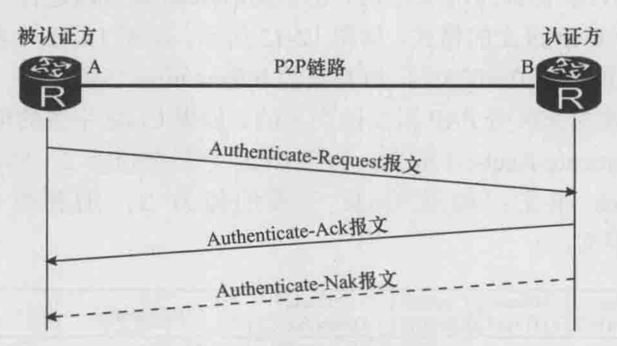


图 12-14 PAP 认证的基本过程

需要指出的是，PAP 认证时，用户名和密码只能以明文形式包含在 Authenticate-Request 报文中，这就存在着很大的安全风险。当被认证方在向认证方发送 PAP 的 Authenticate-Request 报文时，用户名和密码信息很容易被泄露出去。相比之下，CHAP 认证时，相关的认证信息可以以密文的形式包含在 CHAP 报文中，所以其安全性保障程度要远远高于 PAP 认证。

12.1.6　PPP 之网络层协议阶段

如图 12-6 所示，如果 PPP 的 Link Establishment 阶段顺利地结束了，并且 PPP 协议的双方约定无需进行认证，或者双方顺利地结束了认证阶段，那么 PPP 就会自动进入到 Network Layer Protocol 阶段。

在 Network Layer Protocol 阶段，PPP 协议的双方会首先通过 NCP（Network Control Protocol）协议来对网络层协议的参数进行协商，协商一致之后，双方才能够在 PPP 链路上传递携带有相应的网络层协议数据单元的 PPP 帧。

从图 12-5 中我们可以看到，NCP 协议是一个泛称，它包含了许多具体的协议。例如，IPCP（Internet Protocol Control Protocol）协议就是一个 NCP 协议，Novell IPX Control Protocol 也是一个 NCP 协议，如此等等。如果 PPP 链路上需要传递 IP 报文（注：IP 报文是一种网络层协议数据单元），那么 PPP 链路两端的接口就必须通过 IPCP 这个 NCP 先进行协商，协商一致后，PPP 链路上才能传递携带有 IP 报文的 PPP 帧；如果 PPP 链路上需要传递 Novell IPX 报文（注：Novell IPX 报文是一种网络层协议数据单元），那么 PPP 链路两端的接口就必须通过 Novell IPX Control Protocol 这个 NCP 先进行协商，协商一致后，PPP 链路上才能传递携带有 Novell IPX 报文的 PPP 帧；如果 PPP 链路上既需要传递 IP 报文，同时也需要传递 Novell IPX 报文，那么 PPP 链路两端的接口就必须通过 IPCP 进行协商，还必须通过 Novell IPX Control Protocol 进行协商。总之，每一个特定的网络层协议都有一个特定的 NCP 与之相对应。

显然，IP 报文是应用最为广泛的网络层协议数据单元，所以我们接下来简要地描述一下 IPCP 协议。

图 12-15 显示了 IPCP 报文的格式。从图 12-15 中我们可以看到，PPP 帧的 Protocol 字段的值为 0xc8021 时，PPP 帧的 Information 字段便是一个 IPCP 报文。IPCP 报文的 Code 字段是用来区分 IPCP 报文的类型的。IPCP 报文一共有 7 种类型，如果 Code 字段的值为 1，则表明 IPCP 报文是一个 Configure-Request 报文；如果 Code 字段的值为 2，则表明 IPCP 报文是一个 Configure-Ack 报文；如果 Code 字段的值为 3，则表明 IPCP 报文是一个 Configure-Nak 报文，如此等等。

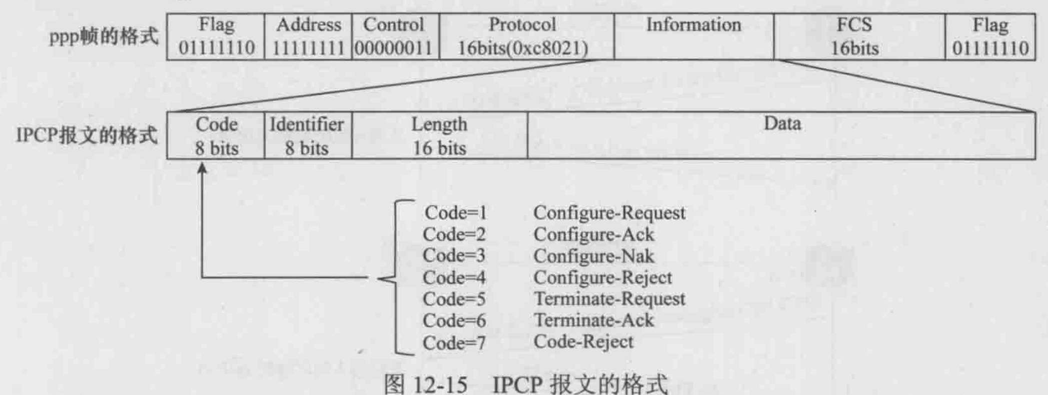


图 12-15　IPCP 报文的格式

如图 12-16 所示，IPCP 主要是通过 Configure-Request 报文和 Configure-Ack 报文来对网络层的 IP 协议进行协商的。因为协商是一个双向过程，所以 PPP 链路的每一个接

口都会向对端接口发送 Configure-Request 报文。当每一个接口都收到了对端接口回应的 Configure-Ack 报文时，才标志着协商取得了一致。IPCP 协商取得了一致之后，PPP 链路上就可以开始传递携带有 IP 报文的 PPP 帧了。注意，此时 PPP 仍然工作在 Network Layer Protocol 阶段。

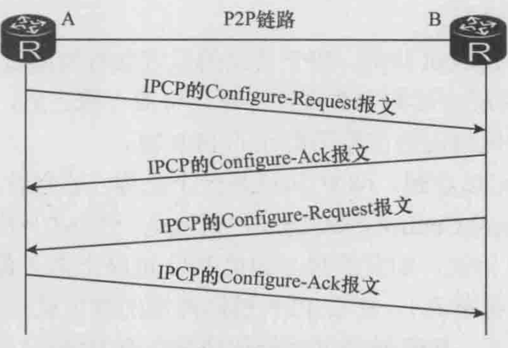


图 12-16　最简形式的 IPCP 协商过程

那么，IPCP 究竟需要协商什么样的内容呢？IPCP 协商的内容主要包括两项，一项是关于 IP 报文的压缩方式，也就协商双方在传递 IP 报文时，IP 报文是采用标准的、非压缩形式的报文格式呢，还是采用压缩形式的报文格式；另一项是关于接口的 IP 地址。我们接下来只对第二项内容进行简单的描述。

如图 12-17 所示，如果网络管理员事先配置了接口 A 的 IP 地址为 IP-A，并且希望对端知道并认可这个 IP 地址，那么在 A 发送的 Configure-Request 报文的配置选项中就应包含 IP-A 这个 IP 地址。B 在接收到来自 A 的 Configure-Request 报文后，会检查 Configure-Request 报文的配置选项中的 IP-A 是否为一个合法的单播 IP 地址，并且是否与自己的 IP 地址发生了冲突。如果 B 发现 IP-A 是一个合法的单播 IP 地址，并且不与自己的 IP 地址发生冲突，那么 B 就会回应一个 Configure-Ack 报文，表示自己知道并认可了 A 的 IP 地址为 IP-A。如果 B 发现 IP-A 不是一个合法的单播 IP 地址，或者 IP-A 与自己的 IP 地址发生了冲突，那么 B 就会回应一个 Configure-Nak 报文，表示自己不认可 A 的 IP 地址为 IP-A，这也意味着 A 需要对 IP-A 进行修改并重新发送 Configure-Request 报文。

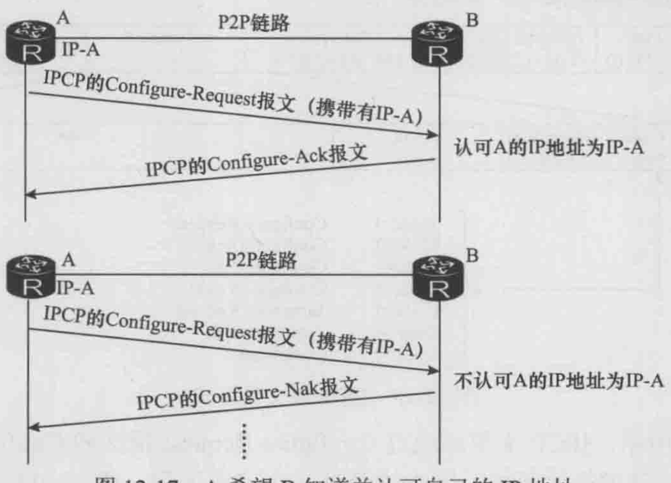


图 12-17　A 希望 B 知道并认可自己的 IP 地址

如图12-18所示，如果网络管理员没有给接口A配置IP地址，而是希望对端设备给接口A分配一个IP地址，那么在A发送的Configure-Request报文的配置选项中就应包含0.0.0.0这个特殊IP地址。B在接收到来自A的Configure-Request报文后，会检查Configure-Request报文的配置选项中的IP地址。B在发现这个IP地址为0.0.0.0时，就会明白对端是在请求从自己这里获取一个IP地址。于是，B就会回应一个Configure-Nak报文，并把自己分配给接口A的IP地址（假设这个IP地址为IP-A）置于Configure-Nak报文的配置选项中。A在接收到来自B的Configure-Nak报文后，会提取出其中的IP-A，然后重新向B发送一个Configure-Request报文，该报文的配置选项中包含了IP-A。B在接收到来自A的Configure-Request报文后，仍然会检查其中的IP-A是否为一个合法的单播IP地址，并且是否与自己的IP地址发生冲突。如果IP-A是一个合法的单播IP地址，并且不与自己的IP地址发生冲突，那么B就会向A发送一个Configure-Ack报文。这样，A就成功地从B那里获得了IP-A这个IP地址。

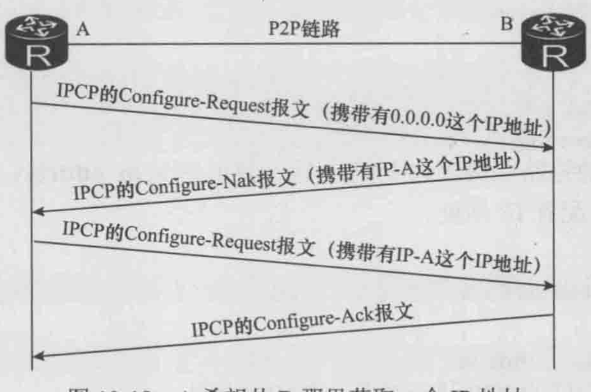


图12-18　A希望从B那里获取一个IP地址

NCP的协商过程与LCP的协商过程有很大的相似性，主要都是通过交互Configure-Request、Configure-Ack、Configure-Nak等报文来完成的，并且所有的配置选项都是包含在同一个Configure-Request报文中一次性进行协商的。如果某些配置选项没有出现在Configure-Request报文中，则表明这些配置选项是取PPP协议规定的“缺省值”。例如，如果IPCP的Configure-Request报文中没有包含关于IP报文的压缩形式这个配置选项，则表示IPCP Configure-Request报文的发送方希望采用的是标准的、非压缩形式的IP报文格式。如果IPCP的Configure-Request报文中没有包含关于IP地址这个配置选项，则表示IPCP Configure-Request报文的发送方不需要IP地址。顺便提一下，路由器上的PPP接口如果没有配置IP地址，通信也是可以正常进行的。

12.1.7 PPP基本配置示例

如图12-19所示，路由器R1和R2通过PPP链路直接相连。网络管理员希望在R1的Serial 0/0/1接口上配置IP地址10.0.0.1/30，在R2的Serial 0/0/1接口上配置IP地址10.0.0.2/30，并且希望R1和R2之间能够传递IP报文。

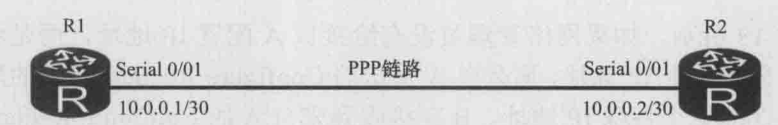


图 12-19 PPP 基本配置示例之一

1. 配置思路

（1）将 R1 的 Serial 0/0/1 接口以及 R2 的 Serial 0/0/1 接口的链路层协议配置为 PPP。

（2）在 R1 的 Serial 0/0/1 接口以及 R2 的 Serial 0/0/1 接口上分别配置 IP 地址。

2. 配置步骤

要在路由器上配置接口的链路层协议为 PPP，必须首先进入到系统视图，然后执行命令 **interface** *interface-type interface-number*，进入指定的接口视图。然后，在接口视图下，执行 **link-protocol ppp** 命令即可将当前接口的链路层协议配置为 PPP。

#配置 R1。

```
<R1> system-view
[R1] interface serial 0/0/1
[R1-Serial0/0/1] link-protocol ppp
```

#配置 R2。

```
<R2> system-view
[R2] interface serial 0/0/1
[R2-Serial0/0/1] link-protocol ppp
```

在配置完接口的链路层协议为 PPP 之后，使用命令 **ip address** *ip-address* { *mask* | *mask-length* }为接口配置 IP 地址。

#配置 R1。

```
[R1-Serial0/0/1] ip address 10.0.0.1 30
```

#配置 R2。

```
[R2-Serial0/0/1] ip address 10.0.0.2 30
```

我们现在对配置好的 PPP 进行确认。以 R1 为例。

```
[R1] display interface serial 0/0/1
Serial0/0/1 current state : UP
Line protocol current state : UP
……
Internet Address is 10.0.0.1/30
Link layer protocol is PPP
LCP opened, IPCP opened
……
```

上面的回显信息中，“Internet Address is 10.0.0.1/30”表示 R1 的 Serial 0/0/1 接口的 IP 地址为 10.0.0.1/30。“Link layer protocol is PPP”表示 R1 的 Serial 0/0/1 接口的数据链路层协议为 PPP。“LCP opened，IPCP opened”表示 LCP 和 IPCP 协商已经成功。注意，既然 NCP 采用的是 IPCP，就说明 PPP 链路上已经可以传递 IP 报文了。

为了验证该 PPP 链路上可以传递 IP 报文，我们可以使用 **Ping** 命令。

```
<R1> ping 10.0.0.2
  PING 10.0.0.2 : 56 data bytes,   press CTRL_C to break
    Reply from 10.0.0.2 : bytes=56 Sequence=1 ttl=255 time=50 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=2 ttl=255 time=50 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=3 ttl=255 time=50 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=4 ttl=255 time=60 ms
```

```
    Reply from 10.0.0.2 : bytes=56 Sequence=5 ttl=255 time=30 ms

  --- 10.0.0.2 ping statistics ---
    5 packet（s）transmitted
    5 packet（s）received
    0.00% packet loss
round-trip min/avg/max = 30/48/60 ms
```

可以看到，显示的结果是与我们的预期完全一致的。

我们再来看一个例子。如图 12-20 所示，路由器 R1 和 R2 通过 PPP 链路直接相连。网络管理员在 R1 的 Serial 0/0/1 接口上配置了 IP 地址 10.0.0.1/30，同时要求 R1 给 R2 的 Serial 0/0/1 接口分配一个 IP 地址 10.0.0.2，并且希望 R1 和 R2 之间能够传递 IP 报文。

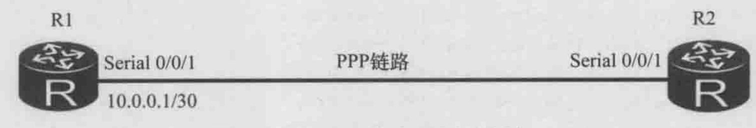


图 12-20　PPP 基本配置示例之二

1. 配置思路

（1）将 R1 的 Serial 0/0/1 接口以及 R2 的 Serial 0/0/1 接口的链路层协议配置为 PPP。

（2）在 R1 的 Serial 0/0/1 接口上配置 IP 地址 10.0.0.1，并指定分配给 R2 的 IP 地址为 10.0.0.2。

（3）将 R2 的 Serial 0/0/1 接口的 IP 地址的获取方式配置为由对端分配的方式。

2. 配置步骤

将 R1 和 R2 的 Serial 0/0/1 接口的链路层协议配置为 PPP。

#配置 R1。

```
<R1> system-view
[R1] interface serial 0/0/1
[R1-Serial0/0/1] link-protocol ppp
```

#配置 R2。

```
<R2> system-view
[R2] interface serial 0/0/1
[R2-Serial0/0/1] link-protocol ppp
```

配置 R1 的 Serial 0/0/1 接口的 IP 地址为 10.0.0.1/30，然后通过命令 **remote address** *ip-address* 命令来指定分配给对端接口（R2 的 Serial 0/0/1 接口）的 IP 地址为 10.0.0.2。

#配置 R1。

```
[R1-Serial0/0/1] ip address 10.0.0.1 30
[R1-Serial0/0/1] remote address 10.0.0.2
```

在 R2 的 Serial 0/0/1 接口视图下执行命令 **ip address ppp-negotiate**，表示希望对端接口分配一个 IP 地址给自己。

#配置 R2。

```
[R2-Serial0/0/1] ip address ppp-negotiate
```

我们现在对所做配置进行验证。以 R2 为例。

```
[R2] display interface serial 0/0/1
Serial0/0/1 current state : UP
```

```
Line protocol current state : UP
......
Internet Address is negotiated, 10.0.0.2/32
Link layer protocol is PPP
LCP opened, IPCP opened
......
```

上面的回显信息中，“Internet Address is negotiated，10.0.0.1/32”表明 R2 的 Serial 0/0/1 接口的 IP 地址为 10.0.0.2/32，该地址是通过协商而获得的，也就是从 R1 那里获得的。

最后，为了验证该 PPP 链路上可以传递 IP 报文，我们可以使用 **Ping** 命令。

```
<R1> ping 10.0.0.2
  PING 10.0.0.2 : 56 data bytes,   press CTRL_C to break
    Reply from 10.0.0.2 : bytes=56 Sequence=1 ttl=255 time=130 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=2 ttl=255 time=10 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=3 ttl=255 time=30 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=4 ttl=255 time=20 ms
    Reply from 10.0.0.2 : bytes=56 Sequence=5 ttl=255 time=30 ms

  --- 10.0.0.2 ping statistics ---
    5 packet（s）transmitted
    5 packet（s）received
    0.00% packet loss
round-trip min/avg/max = 10/44/130 ms
```

可以看到，显示的结果是与我们的预期完全一致的。

12.2 PPPoE

12.2.1 PPPoE 协议的基本概念

我们先来看一下家庭用户上网的一种典型组网场景，如图 12-21 所示。图 12-21 中，PC 1-1、PC 1-2、PC 1-3 以及家庭网关 HG-1（注：HG 是 Home Gateway 的简称）组成了一个家庭网络，在这个家庭网络中，终端 PC 通常是通过常见的标准以太链路或 FE 链路与 HG-1 相连。HG-1 是家庭网络 1 的出口网关路由器。为了利用已经铺设好的电话线路，HG-1 会利用 ADSL（Asymmetric Digital Subscriber Line）技术将自己准备向外发送的以太帧信号调制成一种适合在电话线路上传输的物理信号后再进行发送。网络运营商的 IP-DSLAM（IP Digital Subscriber Line Multiplexer）设备会接收来自不同 HG 的 ADSL 信号，并将其中的以太帧信息解调出来，然后通过一条 GE 链路将这些以太帧送往一个被称为 AC（Access Concentrator）的设备。从数据链路层的角度来看，IP-DSLAM 设备就是一台普通的二层以太网汇聚交换机。

我们知道，网络运营商是要对家庭用户上网进行收费及其他一些接入控制行为的。然而我们也知道，IP-DSLAM 转发给 AC 的帧都是一些以太帧；显然，这些以太帧是无法标示自己是发自 HG-1 的呢，还是发自 HG-2 的呢。从帧的结构上来看，一个以太帧中是没有任何字段可以携带“用户名”和“密码”这些信息的。运营商如果不能区分来自不同的家庭用户的数据流量，当然也就无法进行收费等行为了。

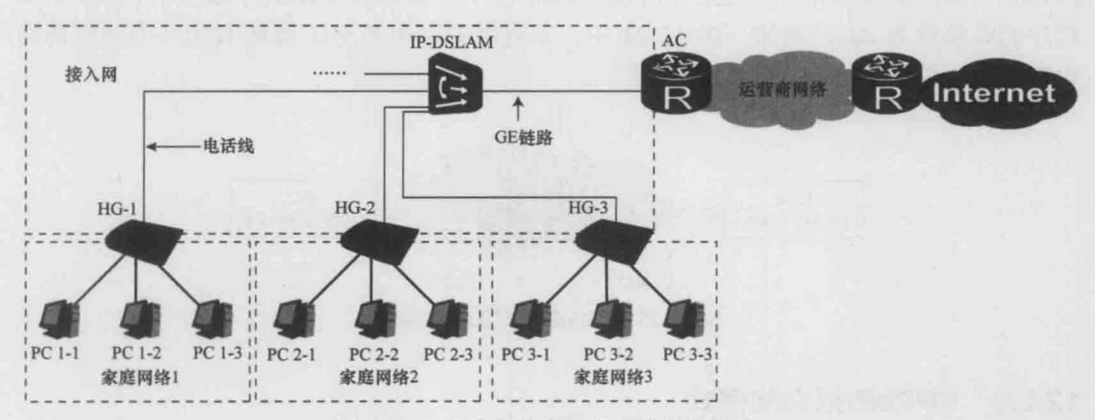


图 12-21　家庭用户上网的一种组网场景

因此，在图 12-21 中，AC 设备必须根据所接收到的以太帧来识别这些帧所对应的家庭用户，并采用用户名和密码的形式来对不同的家庭用户进行认证。在此基础之上，运营商才有可能对家庭用户的上网活动进行计费等管理控制行为。

我们知道，PPP 协议本身就具备了通过用户名和密码的形式进行认证的功能。然而，PPP 协议只适用于点到点的网络类型。图 12-21 中，不同的 HG 和 AC 构成的以太网是一个多点接入网络（Multi-Access Network），因此 PPP 协议无法直接应用在这样的网络上。为了将 PPP 协议应用在以太网上，一种被称为 PPPoE 的协议便应运而生。

从本质上讲，PPPoE（PPP over Ethernet）是一个允许在以太广播域中的两个以太接口之间创建点对点隧道的协议，它描述了如何将 PPP 帧封装在以太帧中。从 PPPoE 的角度来看，图 12-21 中的接入网部分可以简化为图 12-22 所示的网络。

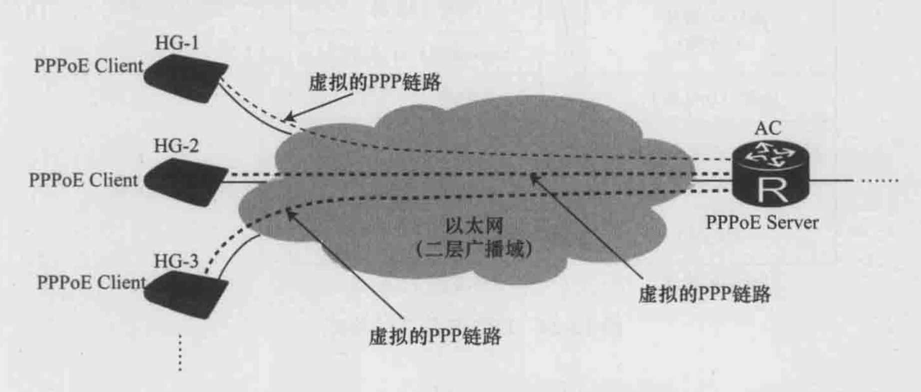


图 12-22　从 PPPoE 的角度看接入网

图 12-22 中，利用 PPPoE 协议，每个家庭用户的 HG 都可以与 AC 之间建立起一条虚拟的 PPP 链路（逻辑意义上的 PPP 链路）。也就是说，HG 与 AC 是可以交互 PPP 帧的。然而，这些 PPP 帧并非是在真实的物理 PPP 链路上传递的，而是被包裹在 HG 与 AC 之间交互的以太帧中，并随这些以太帧在以太链路上的传递而传递的。

图 12-23 显示了 PPPoE 协议的基本架构。PPPoE 协议采用了 Client/Server 模式。在

PPPoE 协议的标准术语中，运行 PPPoE Client 程序的设备称为 Host，运行 PPPoE Server 程序的设备称为 AC。例如，图 12-22 中，家庭网关路由器 HG 就是 Host，而运营商路由器就是 AC。

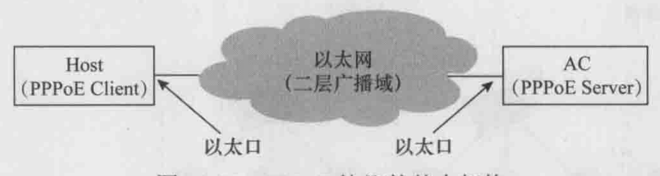


图 12-23 PPPoE 协议的基本架构

12.2.2 PPPoE 报文的格式

图 12-24 显示了 PPPoE 报文的格式。如果以太帧的类型字段的值为 0x8863 或 0x8864，则表明以太帧的载荷数据就是一个 PPPoE 报文。

PPPoE 报文分为 PPPoE Header 和 PPPoE Payload 两个部分。在 PPPoE Header 中，VER 字段（版本字段）的值总是取 0x1，Type 字段的值也总是取 0x1，Code 字段是用来表示不同类型的 PPPoE 报文的，Length 字段用来表示整个 PPPoE 报文的长度，Session-ID 字段用来区分不同的 PPPoE 会话（PPPoE Session）。

到此为止，我们仍不知道 PPP 帧是如何封装在以太帧中的。不用着急，很快我们就会知道，原来 PPP 帧是出现在 PPPoE Payload 中的。

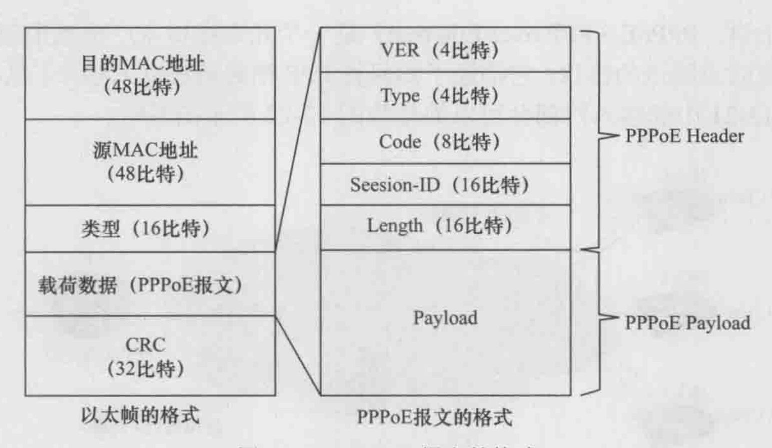


图 12-24 PPPoE 报文的格式

12.2.3 PPPoE 的工作过程

PPPoE 的工作过程分为两个不同的阶段，即 Discovery 阶段（发现阶段）和 PPP Session 阶段（PPP 会话阶段）。

1. Discovery 阶段

如图 12-25 所示，在 PPPoE 发现阶段，Host 与 AC 之间会交互 4 种不同类型的 PPPoE 报文，分别是 PADI（PPPoE Active Discovery Initiation）报文（PPPoE Header 中 Code 字

段的值为0x09)、PADO（PPPoE Active Discovery Offer）报文（PPPoE Header中Code字段的值为0x07)、PADR（PPPoE Active Discovery Request）报文（PPPoE Header中Code字段的值为0x19)、PADS（PPPoE Active Discovery Session-confirmation）报文（PPPoE Header中Code字段的值为0x65)。

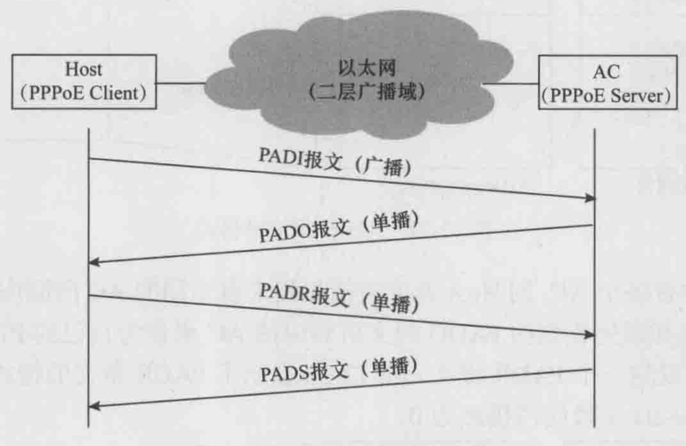


图12-25　PPPoE的发现阶段

首先，Host会以广播方式发送一个PADI报文（见图12-26)，目的是寻找网络中的AC，并告诉AC自己希望获得的服务类型信息。如图12-26所示，在PADI报文的Payload中，包含的是若干个具有Type-Length-Value结构的Tag字段，这些Tag字段表达了Host想要获得的各种服务类型信息。注意，PADI报文中的Session-ID字段的值为0。

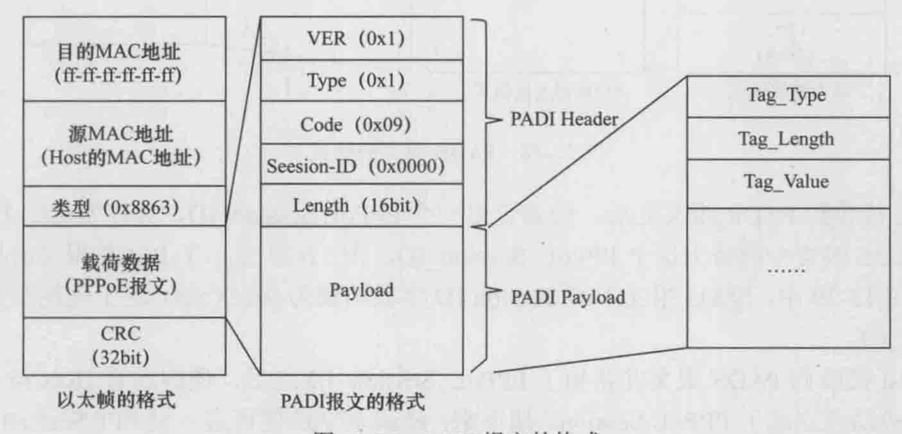


图12-26　PADI报文的格式

AC接收到PADI报文之后，会将PADI报文中所请求的服务与自己能够提供的服务进行比较。AC如果能够提供Host所请求的服务，则单播回复一个PADO报文；如果不能提供，则不做任何回应。图12-27显示了PADO报文的格式。注意，PADO报文中的Session-ID字段的值为0。

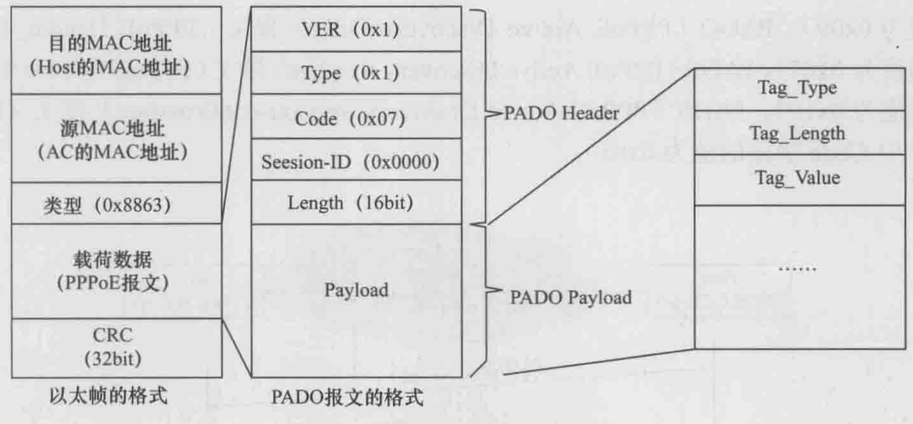


图 12-27　PADO 报文的格式

如果网络中有多个 AC，则 Host 就可能接收到来自不同的 AC 所回应的 PADO 报文。通常，Host 会选择最先收到的 PADO 报文所对应的 AC 来作为自己的 PPPoE Server，并向这个 AC 单播发送一个 PADR 报文。图 12-28 显示了 PADR 报文的格式。注意，PADR 报文中的 Session-ID 字段的值仍然为 0。

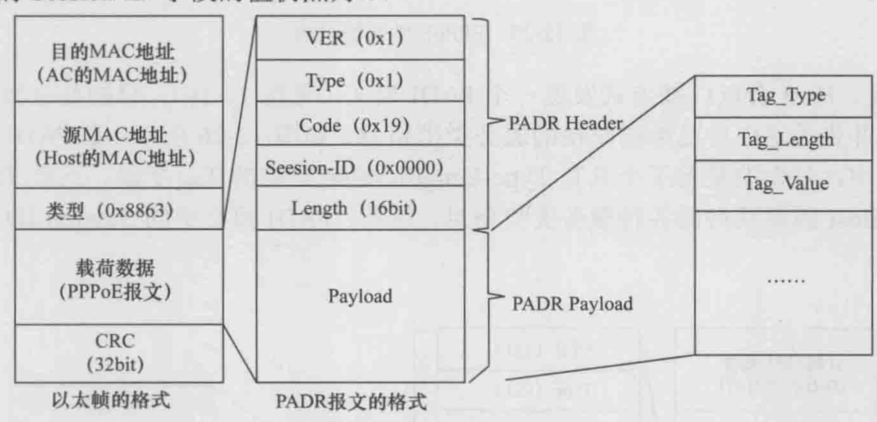


图 12-28　PADR 报文的格式

AC 接收到 PADR 报文之后，会确定出一个 PPPoE Session_ID，并在发送给 Host 的单播 PADS 报文中携带上这个 PPPoE Session_ID。图 12-29 显示了 PADS 报文的格式。注意，图 12-29 中，PADS 报文中的 Session-ID 字段的值为 0xXXXX，这个值便是 PPPoE Session_ID。

Host 接收到 PADS 报文并获知了 PPPoE Session_ID 之后，便标志着 Host 与 AC 之间已经成功建立起了 PPPoE Session。接下来，Host 和 AC 便可进入到 PPP Session 阶段。

2. PPP Session 阶段

在 PPP Session 阶段，Host 与 AC 之间交互的仍然是以太帧，但是这些以太帧中携带了 PPP 帧。图 12-30 显示了在 PPP Session 阶段 Host 与 AC 之间交互的以太帧所包含的内容。从图 12-30 中我们可以看到，以太帧的类型字段的值为 0x8864（注：在 Discovery 阶段，以太帧的类型字段的值总是为 0x8863），表明以太帧的载荷数据仍然是一个 PPPoE

报文。PPPoE 报文中，Code 字段的值取 0x00，Session-ID 字段的值保持为在 Discovery 阶段所确定的值。现在我们终于可以看到，此时的 PPPoE 报文的 Payload 就是一个 PPP 帧！然而，需要注意的是，PPPoE 报文的 Payload 并非是我们之前所熟悉的一个完整的 PPP 帧，而只是 PPP 帧的 Protocol 字段和 Information 字段。之所以如此，是因为 PPP 帧的其他字段在此虚拟的 PPP 链路上已无存在的必要。

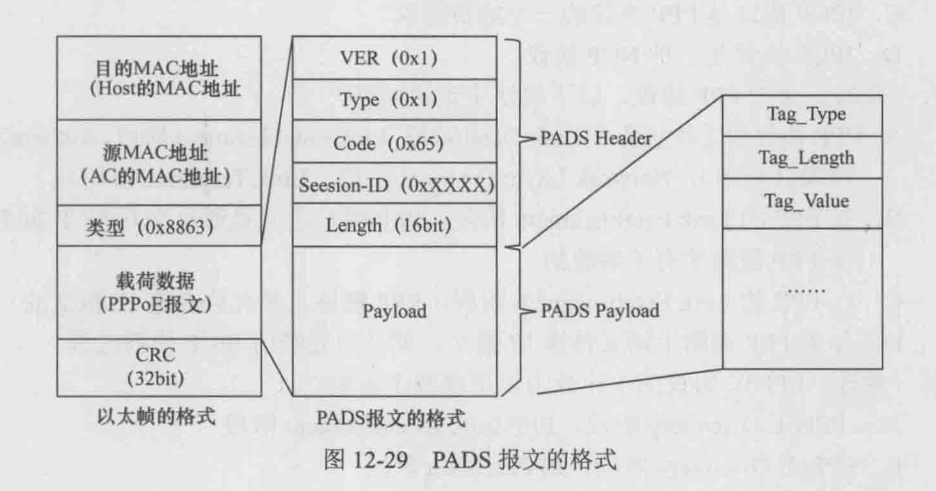


图 12-29　PADS 报文的格式

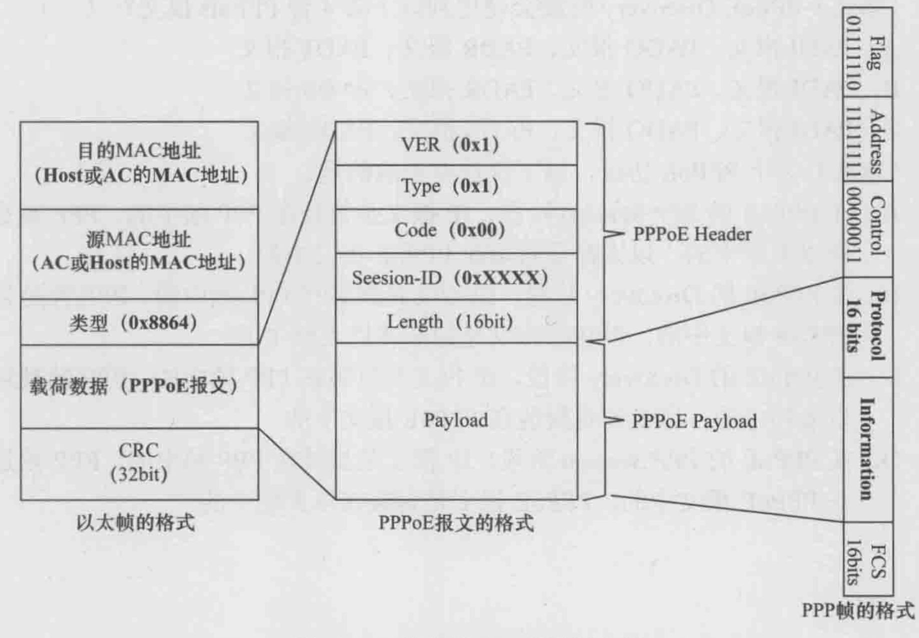


图 12-30　携带有 PPP 帧的以太帧

我们看到，通过 PPPoE 协议的中介作用，在 PPP Session 阶段 Host 与 AC 之间就可以交互 PPP 帧了。通过 PPP 帧的交互，Host 和 AC 便可经历 PPP 的 Link Establishment 阶段，Authentication 阶段以及 Network Layer Protocol 阶段，最终实现 IP 报文的交互。

12.3 练习题

1.（多选）关于 PPP 协议，以下说法中正确的是？（ ）

A．LCP 协议是 PPP 协议的一个成员协议

B．PAP 协议是 PPP 协议的一个成员协议

C．IPCP 协议是 PPP 协议的一个成员协议

D．IPCP 协议是一种 NCP 协议

2.（多选）关于 PPP 协议，以下说法中正确的是？（ ）

A．PPP 协议的工作包含了 Link Dead 阶段，Link Establishment 阶段，Authentication 阶段（可选），Network Layer Protocol 阶段，Link Termination 阶段

B．在 PPP 的 Link Establishment 阶段，PPP 接口之间是通过交互 NCP 报文来协商 PPP 链路的有关参数的

C．在 PPP 的 Link Establishment 阶段，PPP 链路上是允许传递 IP 报文的

D．如果 PPP 链路上需要传递 IP 报文，则必须先经历 IPCP 协商过程

3.（单选）PPPoE 协议的工作分为以下哪两个阶段？（ ）

A．PPPoE Discovery 阶段，PPP Link Establishment 阶段

B．PPPoE Discovery 阶段，PPP Session 阶段

C．PPPoE Discovery 阶段，PPPoE Authentication 阶段

4.（单选）PPPoE Discovery 阶段会使用到以下哪 4 种 PPPoE 报文？（ ）

A．PADI 报文、PADO 报文、PADR 报文、PADT 报文

B．PADI 报文、PADO 报文、PADR 报文、PADS 报文

C．PADI 报文、PADO 报文、PADS 报文、PADT 报文

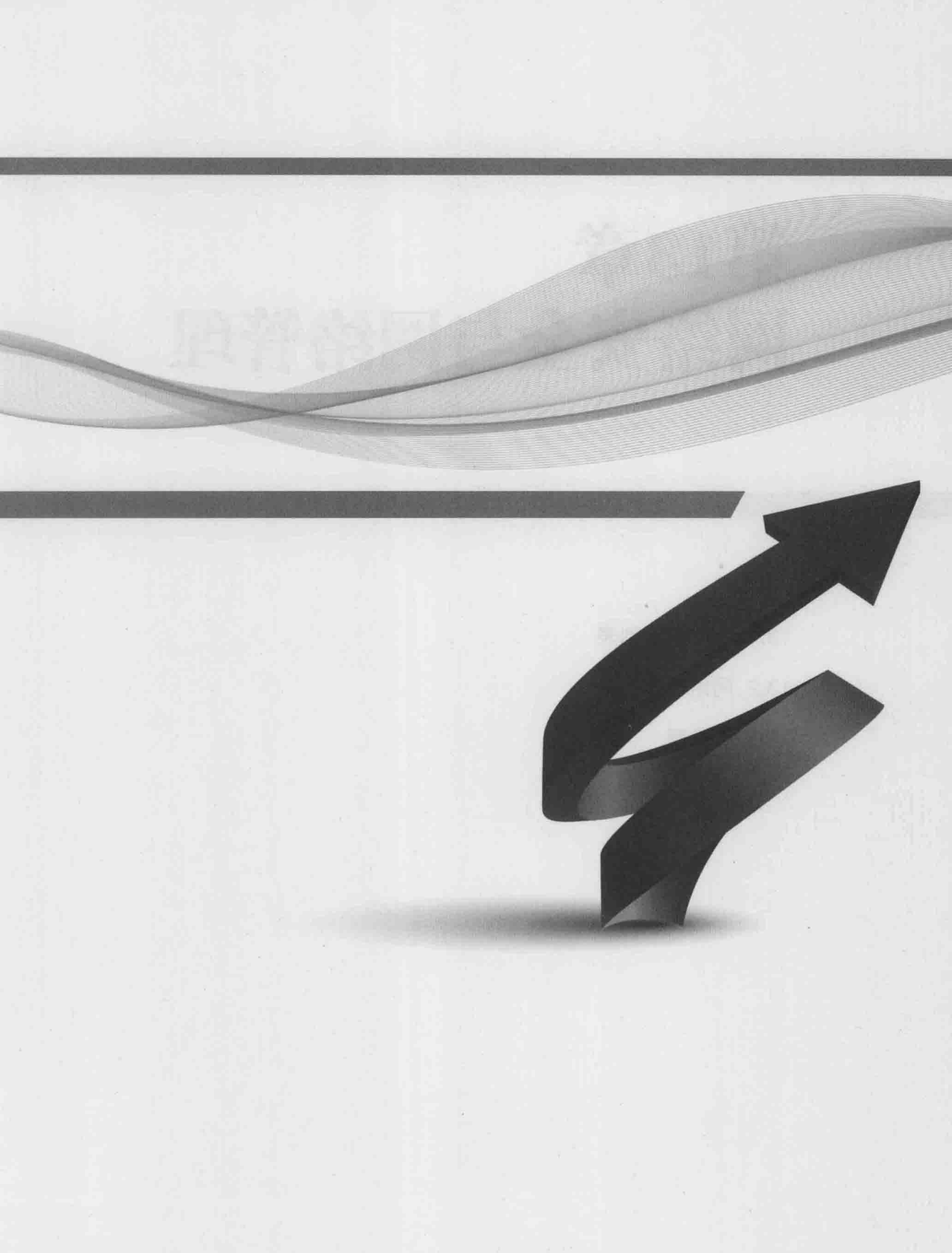
5.（单选）关于 PPPoE 协议，以下说法中正确的是？（ ）

A．在 PPPoE 的 PPP Session 阶段，IP 报文是封装在 PPP 帧中的，PPP 帧是封装在以太帧中的，以太帧是封装在 PPPoE 报文中的

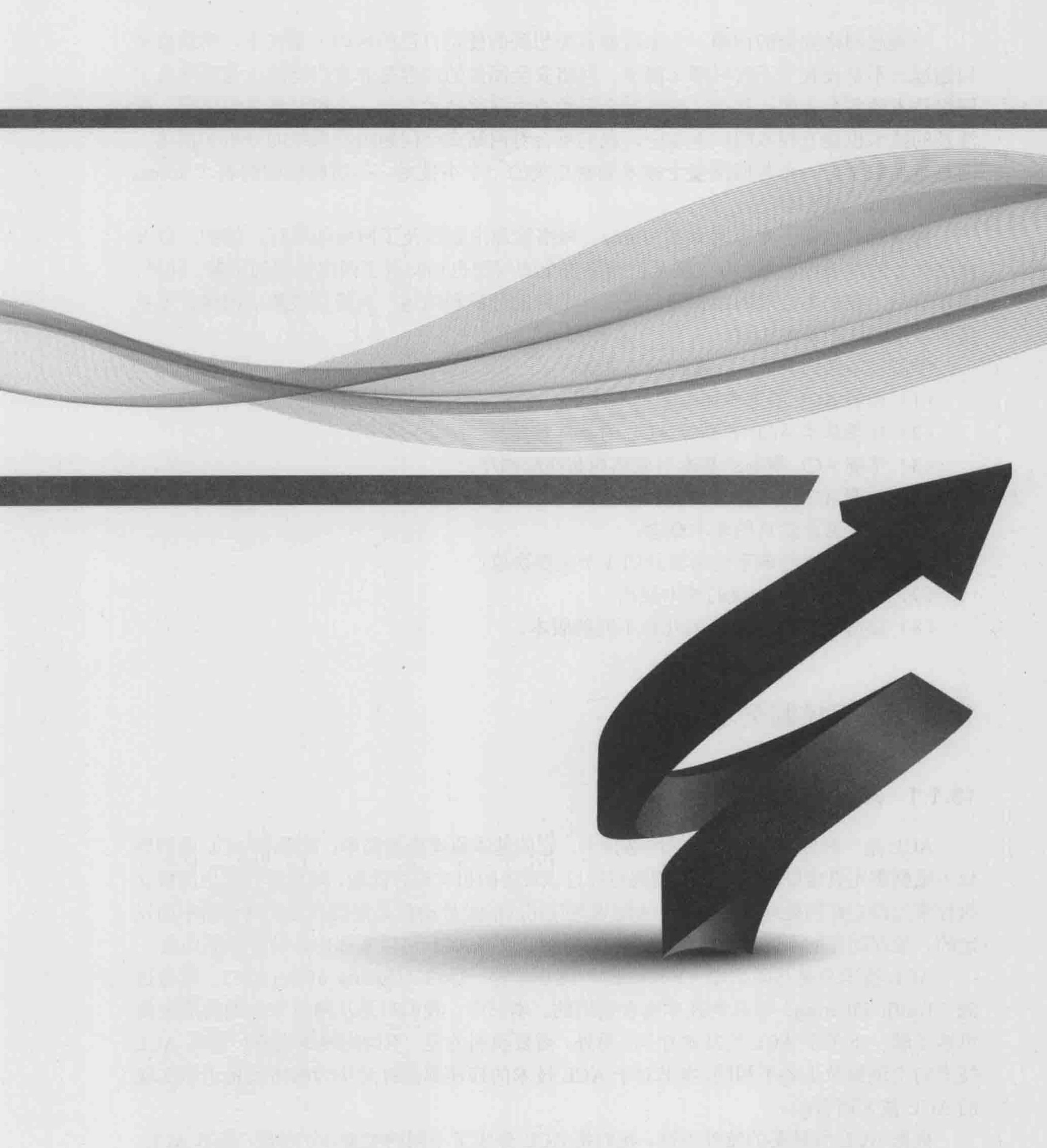
B．在 PPPoE 的 Discovery 阶段，IP 报文是封装在 PPP 帧中的，PPP 帧是封装在 PPPoE 报文中的，PPPoE 报文是封装在以太帧中的

C．在 PPPoE 的 Discovery 阶段，IP 报文是封装在 PPP 帧中的，PPP 帧是封装在以太帧中的，以太帧是封装在 PPPoE 报文中的

D．在 PPPoE 的 PPP Session 阶段，IP 报文是封装在 PPP 帧中的，PPP 帧是封装在 PPPoE 报文中的，PPPoE 报文是封装在以太帧中的



第13章 网络安全与网络管理



一提起网络安全的问题，大家可能首先想到的便是自己的密码。事实上，网络安全问题远远不是设置一下密码那么简单。网络安全涵盖的内容是非常广泛的，几乎涉及了网络技术的各个方面。目前，网络安全已经成为网络技术中的一个相对独立的领域，所涉及的技术也是五花八门。本章中，我们不会对网络安全问题进行系统的分析和描述，而只是抽取学习一个与网络安全技术紧密相关的一个小技术——访问控制列表（Access Control List，ACL）。

网络管理也是一个非常大的 Topic，网络管理主要涉及了网络的运行、维护，以及网络业务的部署等问题，有人甚至把网络安全的问题也划归进了网络管理的范畴。同样，在本章中，我们不会对网络管理问题进行系统的分析和描述，而是只聚焦于网络管理系统所涉及的 3 个基本协议。

学习完本章内容之后，我们应该能够：

（1）理解 ACL 的基本原理和基本作用；

（2）熟悉基本 ACL 和高级 ACL 的基本差异；

（3）掌握 ACL 规则的基本组成结构和匹配顺序；

（4）掌握 ACL 中通配符的使用方法；

（5）了解网络管理的基本概念；

（6）了解网络管理系统所涉及的 3 个主要协议；

（7）理解 SNMP 协议的基本架构；

（8）知道 SNMP 所经历的几种不同的版本。

13.1 访问控制列表

13.1.1 ACL 的基本原理

ACL 是一种应用非常广泛的网络技术，它的基本原理极为简单：配置了 ACL 的网络设备根据事先设定好的报文匹配规则对经过该设备的报文进行匹配，然后对匹配上的报文执行事先设定好的处理动作。这些匹配规则及相应的处理动作是根据具体的网络需求而设定的。处理动作的不同以及匹配规则的多样性，使得 ACL 可以发挥出各种各样的功效。

ACL 技术总是与防火墙（Firewall）、路由策略、QoS（Quality of Service）、流量过滤（Traffic Filtering）等其他技术结合使用的。本书中，我们只是从网络安全的角度来简单地了解一下关于 ACL 的基本知识。另外，需要说明的是，不同的网络设备厂商在 ACL 技术的实现细节上各不相同，本书对于 ACL 技术的描述都是针对华为网络设备上所实现的 ACL 技术而言的。

根据 ACL 所具备的特性不同，我们将 ACL 分成了不同的类型，分别是：基本 ACL、高级 ACL、二层 ACL、用户自定义 ACL，其中应用最为广泛的是基本 ACL 和高级 ACL。在网络设备上配置 ACL 时，每一个 ACL 都需要分配一个编号，称为 ACL 编号。基本 ACL、高级 ACL、二层 ACL、用户自定义 ACL 的编号范围分别为 2 000~2 999、3 000~3 999、4 000~4 999、5 000~5 999。配置 ACL 时，ACL 的类型应该与相应的编号

范围保持一致。

一个 ACL 通常由若干条“deny | permit”语句组成，每条语句就是该 ACL 的一条规则，每条语句中的 deny 或 permit 就是与这条规则相对应的处理动作。处理动作 permit 的含义是“允许”，处理动作 deny 的含义是“拒绝”。特别需要说明的是，ACL 技术总是与其他技术结合在一起使用的，因此，所结合的技术不同，“允许（permit）”及“拒绝（deny）”的内涵及作用也会不同。例如，当 ACL 技术与流量过滤技术结合使用时，permit 就是“允许通行”的意思，deny 就是“拒绝通行”的意思。

配置了 ACL 的设备在接收到一个报文之后，会将该报文与 ACL 中的规则逐条进行匹配。如果不能匹配上当前这条规则，则会继续尝试去匹配下一条规则。一旦报文匹配上了某条规则，则设备会对该报文执行这条规则中定义的处理动作（permiet 或 deny），并且不再继续尝试与后续规则进行匹配。如果报文不能匹配上 ACL 的任何一条规则，则设备会对该报文执行 permit 这个处理动作。

一个 ACL 中的每一条规则都有一个相应的编号，称为规则编号（rule-id）。缺省情况下，报文总是按照规则编号从小到大的顺序与规则进行匹配。缺省情况下，设备会在创建 ACL 的过程中自动为每一条规则分配一个编号。如果将规则编号的步长设定为 10（注：规则编号的步长的缺省值为 5），则规则编号将按照 10、20、30、40……这样的规律自动进行分配。如果将规则编号的步长设定为 2，则规则编号将按照 2、4、6、8……这样的规律自动进行分配。步长的大小反映了相邻规则编号之间的间隔大小。间隔的存在，实际上是为了便于在两个相邻的规则之间插入新的规则。

13.1.2　基本 ACL

ACL 分为基本 ACL 和高级 ACL 等类型。基本 ACL 只能基于 IP 报文的源 IP 地址、报文分片标记和时间段信息来定义规则。

配置基本 ACL 规则的命令具有如下的结构。

rule [*rule-id*] {**deny** | **permit**} [**source** {*source-address source-wildcard* | **any**} | **fragment** | **logging** | **time-range** *time-name*]

命令中各个组成项的解释如下。

rule：表示这是一条规则。

rule-id：表示这条规则的编号。

deny | **permit**：这是一个二选一选项，表示与这条规则相关联的处理动作。**deny** 表示“拒绝”；**permit** 表示“允许”。

source：表示源 IP 地址信息。

source-address：表示具体的源 IP 地址。

source-wildcard：表示与 *source-address* 相对应的通配符。*source-wildcard* 和 *source-address* 的结合使用，可以确定出一个 IP 地址的集合。极端情况下，该集合中可以只包含一个 IP 地址。通配符 *source-wildcard* 的使用方法，是与 8.3.11 小节中 *wildcard-mask* 的使用方法完全一样的，所以这里不再赘述。

any：表示源 IP 地址可以是任何地址。

fragment：表示该规则只对非首片分片报文有效。

logging：表示需要将匹配上该规则的 IP 报文进行日志记录。

time-range *time-name*：表示该规则的生效时间段为 *time-name*，具体的使用方法这里不做描述。

如图 13-1 所示，某公司网络包含了研发部区域，人力资源部区域和财务部区域。在研发部区域中，有一台专门供实习人员使用的 PC，该 PC 的 IP 地址是 172.16.10.100/24。出于网络安全方面的考虑，我们需要禁止财务部区域接收到实习人员发送的 IP 报文。为了满足这样的网络需求，我们可以在路由器 R 上配置基本 ACL。基本 ACL 可以根据源 IP 地址信息识别出实习人员发出的 IP 报文，然后在 GE1/0/3 接口的出方向（Outbound 方向）上拒绝放行这样的 IP 报文。

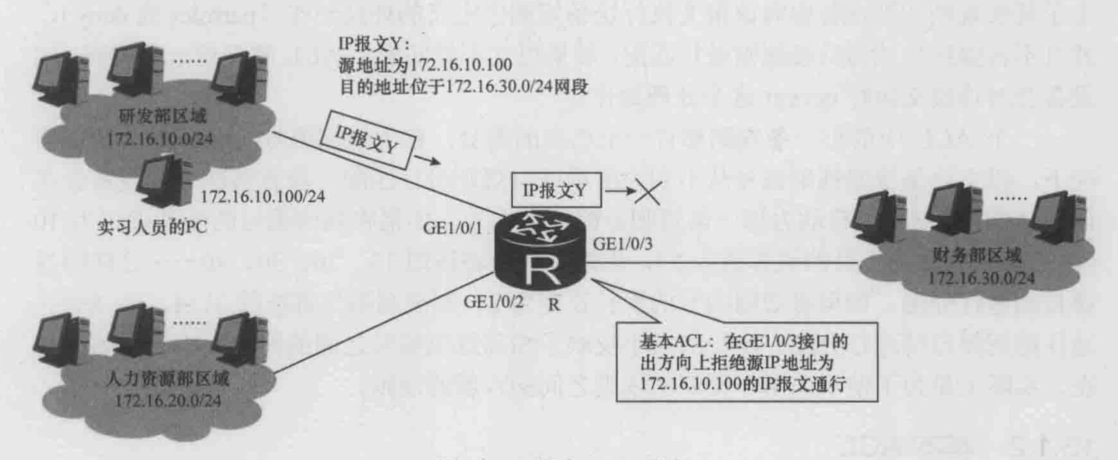


图 13-1 基本 ACL 示意

针对图 13-1 所示的例子，我们来看看应该如何配置路由器 R。首先，我们在 R 的系统视图下创建一个编号为 2000 的 ACL。

```
[R] acl 2000
[R-acl-basic-2000]
```

然后，在 ACL 2000 的视图下创建如下的规则。

```
[R-acl-basic-2000] rule deny source 172.16.10.100  0.0.0.0
[R-acl-basic-2000]
```

这条规则的含义是：拒绝源 IP 地址为 172.16.10.100 的 IP 报文。

最后，使用报文过滤技术中的 **traffic-filter** 命令将 ACL 2000 应用在 R 的 GE1/0/3 接口的出方向上。

```
[R-acl-basic-2000] quit
[R] interface gigabitethernet 1/0/3
[R-GigabitEthernet1/0/3] traffic-filter outbound acl 2000
[R-GigabitEthernet1/0/3]
```

通过上面的配置，源 IP 地址为 172.16.10.100 的 IP 报文便无法在出方向上通过 R 的 GE1/0/3 接口，这样就实现了我们的安全策略。

13.1.3 高级 ACL

高级 ACL 可以根据 IP 报文的源 IP 地址、IP 报文的目的 IP 地址、IP 报文的协议字

段的值、IP 报文的优先级的值、IP 报文的长度值、TCP 报文的源端口号、TCP 报文的目的端口号、UDP 报文的源端口号、UDP 报文的目的端口号等信息来定义规则。基本 ACL 的功能只是高级 ACL 的功能的一个子集，高级 ACL 可以比基本 ACL 定义出更精准、更复杂、更灵活的规则。

高级 ACL 中规则的配置比基本 ACL 中规则的配置要复杂得多，且配置命令的格式也会因 IP 报文的载荷数据的类型不同而不同。例如，针对 ICMP 报文、TCP 报文、UDP 报文等不同类型的报文，其相应的配置命令的格式也是不同的。下面是针对所有 IP 报文的一种简化了的配置命令的格式。

rule [*rule-id*] { **deny** | **permit** } **ip** [**destination** {*destination-address destination-wildcard* | **any**}] [**source** { *source-address source-wildcard* | **any** }]

如图 13-2 所示，该网络的结构与图 13-1 所示的网络完全一样，所不同的是，我们要求实习人员无法接收到来自财务部区域的 IP 报文。在这种情况下，我们可以在路由器 R 上配置高级 ACL。高级 ACL 可以根据目的 IP 地址信息识别出去往实习人员的 IP 报文，然后在 GE1/0/03 接口的入方向（Inbound 方向）上拒绝放行这样的 IP 报文。

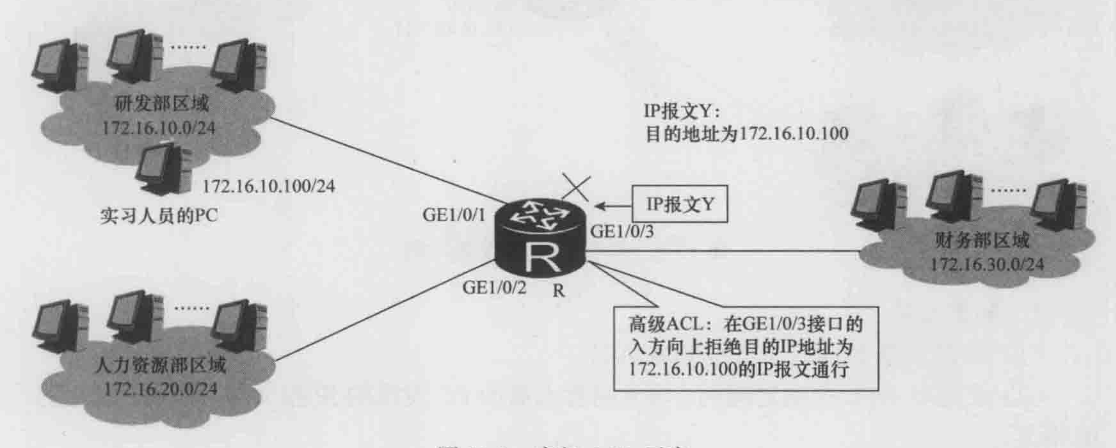


图 13-2　高级 ACL 示意

针对图 13-2 所示的例子，我们来看看应该如何配置路由器 R。首先，我们在 R 的系统视图下创建一个编号为 3000 的 ACL。

```
[R] acl 3000
[R-acl-adv-3000]
```

然后，在 ACL 3000 的视图下创建如下的规则。

```
[R-acl-adv-3000] rule deny destination 172.16.10.100   0.0.0.0
[R-acl-adv-3000]
```

这条规则的含义是：拒绝目的 IP 地址为 172.16.10.100 的 IP 报文。

最后，使用报文过滤技术中的 **traffic-filter** 命令将 ACL 3000 应用在 R 的 GE1/0/3 接口的入方向上。

```
[R-acl-adv-3000] quit
[R] interface gigabitethernet 1/0/3
[R-GigabitEthernet1/0/3] traffic-filter inbound acl 3000
[R-GigabitEthernet1/0/3]
```

通过上面的配置，目的 IP 地址为 172.16.10.100 的 IP 报文便无法在入方向上通过 R 的 GE1/0/3 接口，这样就实现了我们的安全策略。

13.1.4 基本 ACL 的配置示例

图 13-3 显示了某公司的网络结构。出于网络安全方面的考虑，我们希望只有网络管理员的 PC 才能通过 Telnet 方式登录到路由器 Router 上，其他 PC 都不能通过 Telnet 方式登录到路由器。

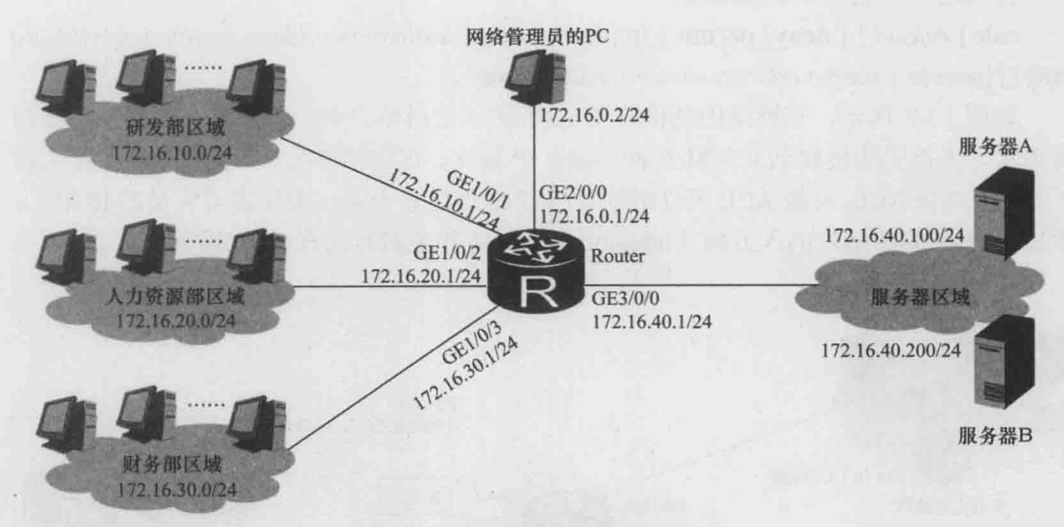


图 13-3 基本 ACL 的配置示例

1. 配置思路

（1）在路由器 Router 上创建基本 ACL。

（2）在基本 ACL 中制定规则，区分网管人员的 PC 发出的 IP 报文与其他 PC 发出的 IP 报文。

（3）在 VTY（Virtual Type Terminal）上应用所配置的基本 ACL。

2. 配置步骤

要在路由器 Router 上创建 ACL，必须首先进入系统视图，然后执行命令 **acl** *acl-number*。对于基本 ACL，*acl-number* 的值必须在 2 000～2 999 的范围内，我们这里确定为 2 000。另外，我们假设网管人员的 PC（IP 地址为 172.16.0.2）已经使用 Telnet 方式登录上了路由器 Router。

#配置路由器 Router。

```
<Router> system-view
[Router] acl 2000
[Router-acl-basic-2000]
```

创建了基本 ACL 2000 后，我们便可以使用 **rule** 命令来制定相应的规则。首先，我们制定一条规则，其含义是允许（permit）源 IP 地址为 172.16.0.2 的 IP 报文。

#配置路由器 Router。

```
[Router-acl-basic-2000] rule permit source 172.16.0.2 0
```

然后，我们再制定一条规则，其含义是拒绝（deny）源 IP 地址为任意地址的 IP 报文。

#配置路由器 Router。

```
[Router-acl-basic-2000] rule deny source any
```

制定完规则之后，我们可以使用 **display acl** 2000 命令来查看 ACL 2000 的配置信息。

```
[Router-acl-basic-2000] quit
[Router] quit
<Router> display acl 2000
Basic ACL 2000, 2 rules
ACL's step is 5
 rule 5 permit source 172.16.0.2 0 （0 times matched）
 rule 10 deny （0 times matched）
```

从回显信息中我们可以看到，基本 ACL 2000 中已经存在两条规则，路由器 Router 为这两条规则自动分配的规则编号分别是 5 和 10。另外，回显信息中的“ACL's step is 5”表示该 ACL 的规则编号的步长为 5，两个“0 times matched”表示该 ACL 的两条规则的匹配次数都为 0（这是因为我们还没有把这个 ACL 应用到路由器上）。

接下来，我们在 VTY 上应用 ACL 2000。

#配置路由器 Router。

```
<Router> system-view
[Router] user-interface vty 0 4
[Router-ui-vty0-4]
[Router-ui-vty0-4] acl 2000 inbound
```

为了确认配置是否生效，我们先退出本次网管人员的 Telnet 登录，然后重新使用 Telnet 登录路由器，发现可以正常登录。

```
<PC> telnet 172.16.0.1
Trying 172.16.0.1 ...
Press CTRL+K to abort
Connected to 172.16.0.1 ...
Info : The max number of VTY users is 10,  and the number of current VTY users on line is 1.
The current login time is 2014-10-03 02:06 : 00.
<Router>
```

在路由器上重新查看 ACL 2000 的配置信息如下。

```
<Router> display acl 2000
Basic ACL 2000, 2 rules
ACL's step is 5
 rule 5 permit source 172.16.0.2 0（1 times matched）
 rule 10 deny（0 times matched）
```

从回显信息中我们可以看到，第一条规则的匹配次数已经变为 1，这说明网管人员的 PC 所发出的 IP 报文已经匹配上了这条规则。

然后，我们在其他某台 PC 上使用 Telnet 方式登录路由器，发现无法正常登录。

```
<PC> telnet 172.16.10.1
Trying 172.16.10.1 ...
Press CTRL+K to abort
Error : Failed to connect to the remote host.
```

再次在路由器上查看 ACL 2000 的配置信息。

```
<Router> display acl 2000
Basic ACL 2000, 2 rules
ACL's step is 5
```

```
rule 5 permit source 172.16.0.2 0 (1 times matched)
rule 10 deny (1 times matched)
```

从回显信息中我们可以看到，第二条规则的匹配次数也变为了 1，这说明刚才尝试 Telnet 登录的那台 PC 所发出的 IP 报文匹配上了第二条规则，但该报文被直接丢弃了。

13.2 网络管理

13.2.1 网络管理的基本概念

首先，我们需要对网络管理有一个基本而直观的认识。我们将以一个公司的办公网络的情况来说明一下网络管理的基本概念。

我们知道，一个规模足够大的公司通常都会建立起自己的办公网络。假设某个公司有几千名员工，其办公网络包含了几台大型服务器，几十台路由器和上百台交换机。针对这个公司的办公网络，我们现在提出这样一些问题。

（1）这个办公网络中，总共究竟有多少台路由器？每台路由器的名称是什么？每台路由器的安放位置在哪里？

（2）此时此刻，出现故障的交换机有几台？都是一些什么样的故障？

（3）“财务部 1 号”路由器与“市场部 2 号”路由器之间的链路此刻是否是断开的？

（4）“财务部 1 号”路由器的 GE1/0/0 接口从昨天中午 12 点到今天中午 12 点这段时间一共接收了多少个 IP 报文？

（5）……

显然，公司的普通员工是无法回答这些问题的。实质上，这些问题都是一些关于网络管理的典型问题。所谓网络管理，简单地讲，就是指在各个层次上对于网络的组成结构和运行状态及时而准确的认识和干预。网络管理是保障网络可靠运行的重要手段。

然而，上面的那些提问对于公司的网络管理人员（简称网管员）来说就可能显得非常简单了。那么，网管员凭什么就能够回答那些问题呢？原来，网管员的电脑屏幕上可以实时地显示出公司办公网络的拓扑图（见图 13-4），这个拓扑图正是公司办公网络的真实写照！拓扑图中的各种设备（如路由器、交换机、服务器等）的图标都是与真实设备一一对应的（但一般不会包含网络中的普通 PC）。如果用鼠标去单击某个设备图标，则图标边上会自动显示出该设备的名称（如“财务部 1 号”）以及该设备的安放位置（如“××楼××层×××房间”）。设备在正常工作时，拓扑图中该设备的图标是某种颜色，当设备出现故障时，其图标就自动变成了另一种颜色。如果我们单击“财务部 1 号”路由器图标并进行少许的操作，屏幕上就会显示出该路由器的 GE1/0/0 接口从昨天中午 12 点到今天中午 12 点这段时间一共接收到了多少个 IP 报文。总之，网管员有了如此神奇的电脑，要迅速而正确地回答那些问题其实是轻易而举的事情。

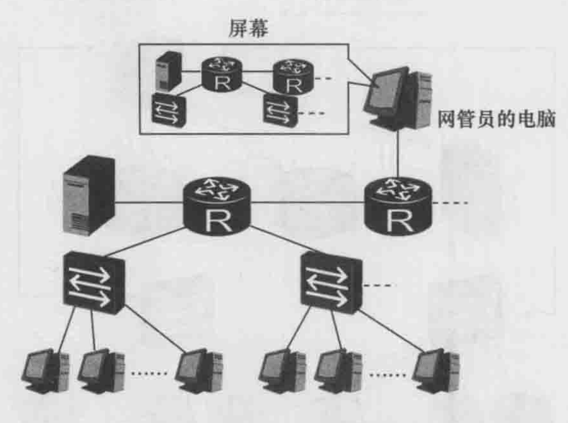


图 13-4　网管员的电脑屏幕

13.2.2　网络管理系统

网管员的电脑之所以如此神奇，是因为网管员在自己的电脑上安装并运行了某种通称为“网络管理系统”的软件工具。例如，华为的 eSight 就是这样一种功能强大的网络管理系统。eSight 是华为推出的专门针对企业网络及数据中心的新一代网络管理系统。网络管理系统一般具有如下的功能。

（1）网络拓扑图的显示。

（2）网络设备端口状态的监视与分析。

（3）网络性能数据的监视与分析。

（4）故障的报警与诊断。

（5）远程配置。

（6）……

为了让网络管理系统正常地工作，网管员除了需要在自己的电脑上安装并运行网管软件（如 eSight）之外，还需要在需要被管理的各个设备上进行一些简单的配置操作。此后，这些设备就能够与网管员的电脑之间进行管理信息的交流（见图 13-5）。网管员的电脑在收集、分析和处理来自各个设备的管理信息之后，便能以图像、表格、文字、甚至声音等形式将网络的各种情况呈现给网络管理人员。

显然，网管员的电脑与被管理的各个设备之间必须通过某种“语言”才能进行管理信息的交流，这种“语言”就是 Simple Network Management Protocol（简单网络管理协议），简称 SNMP 协议。如同 DHCP 协议一样，SNMP 也是一种 Client/Server 模式的网络协议。需要注意的是，运行在网管员的电脑上的是 SNMP Client，而运行在被管理设备上的是 SNMP Server，如图 13-6 所示。从根本上讲，管理信息的交流是通过在 SNMP Server 和 SNMP Client 之间进行 SNMP 报文的交互而实现的。

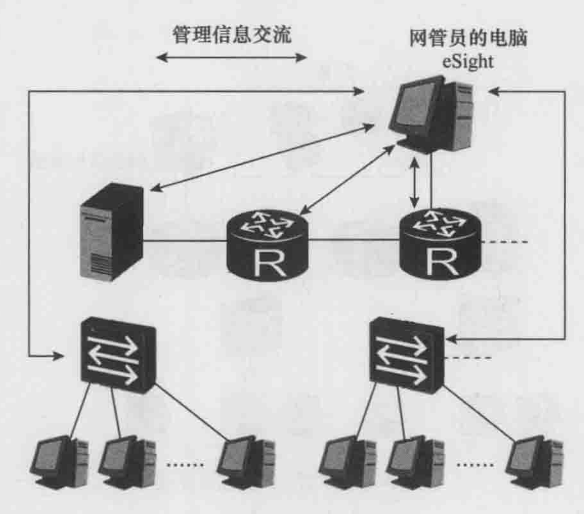


图 13-5 管理信息交流

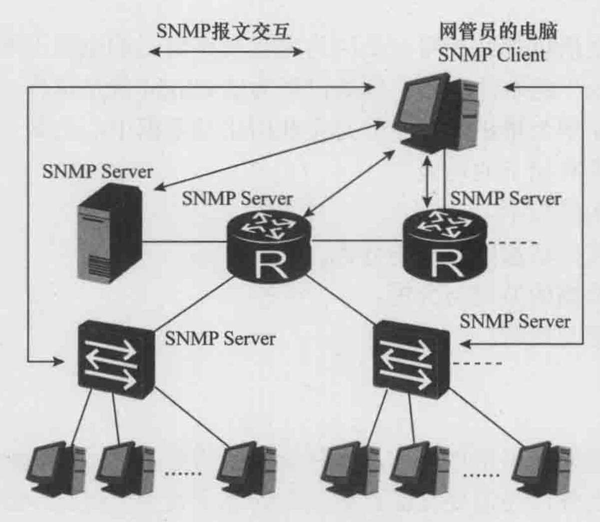


图 13-6 SNMP 管理信息交流

实际上，对于 TCP/IP 网络来说，网络管理系统总共涉及了 3 个协议，分别是 SNMP 协议、SMI（Structure of Management Information，管理信息结构）协议和 MIB（Management Information Base，管理信息库）协议。在这 3 个协议中，SNMP 是核心协议，但 SNMP 协议需要用到其他两个协议。下面，我们就简单地了解一下关于这 3 个协议的基本知识。

13.2.3 SMI 协议

任何时候，当我们谈及"管理"一词的时候，总是会关心什么才是被管理的对象（Object）。网络管理中，被管理的对象可以是一台路由器，可以是一台交换机，可以是路由器上的某一块板卡，也可以是这块板卡上的某一个端口，还可以是这个端口所接收到的 IP 报文的总的数量，如此等等。总之，这些被管理的对象林林总总，数量众多，种类繁杂，并且可以具有不同的层级。

显然，要实现对网络的规范管理，前提之一就是必须事先对被管理的对象进行规范化处理，而这正就是 SMI 协议的内容。概括地讲，SMI 协议的主要内容就是定义了一列的规则，这些规则分为三个方面：第一方面的规则是关于应该如何给被管理对象命名，即对象的命名规则；第二方面的规则是定义了被管理对象有哪些类型，即对象的类型规则；第三方面的规则是关于如何对被管理对象的各种信息进行编码，即对象的编码规则。下面，我们只简单地介绍一下关于被管理对象的命名规则。

在日常生活中，当听说某某人名叫张什么或李什么时，我们就知道此人应该是一个汉族人；当听说某某人名叫叶赫拉拉什么什么或爱新觉罗什么什么时，我们就知道此人应该是一个满族人。也就是说，汉人应该有汉人的名字，满人应该有满人的名字。在 SMI 中，情况也非常类似。SMI 采用了 Object Identifer 来区分不同种类的对象名称，相当于说，汉人名对应了一个 Object Identifier，满人名对应了另外一个不同的 Object Identifer。如图 13-7 所示，SMI 采用了一种树状结构来定义不同的 Object Identifer。

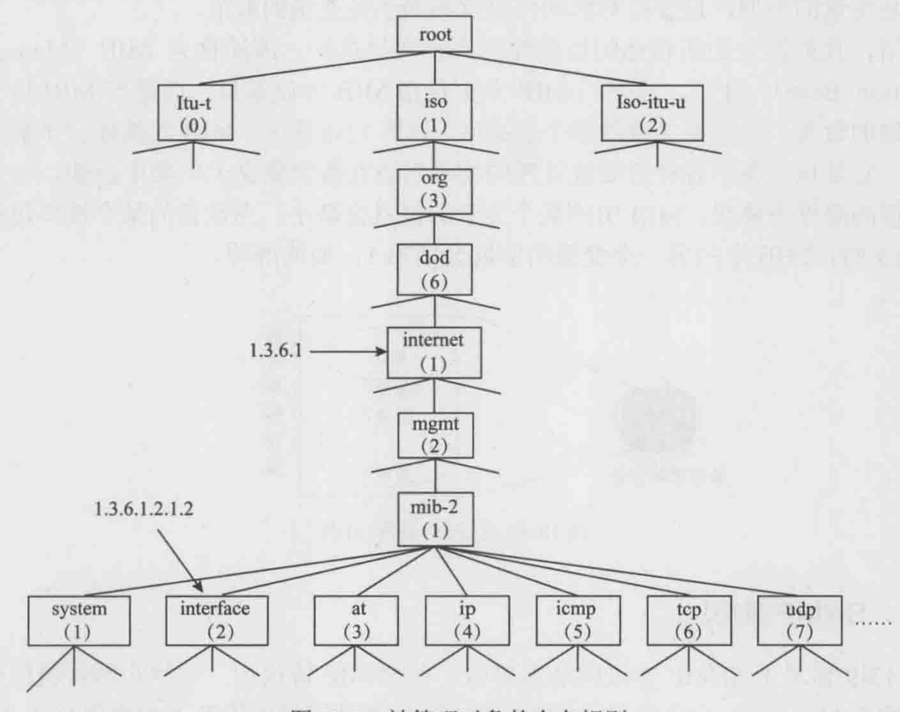


图 13-7　被管理对象的命名规则

图 13-7 中，每一个方框就代表了一种对象（Object），一种对象可以有自己的父对象（Parent Object），同时还可以有若干种子对象（Children Object），由此我们可以看出，对象是有着层次结构的。对于“internet”这种对象，其 Object Identifier 就是 1.3.6.1（iso.org.dod.internet）；对于“interface”这种对象，其 Object Identifier 就是 1.3.6.1.2.1.2（iso.org.dod.internet.mgmt.mib-2.interface）。

还记得前面提到的“财务部 1 号”路由器吗？还记得前面提到的“××楼××层×××房间”吗？这些信息涉及了设备名称（sysName）和设备位置。如果我们把图 13-7 中的树状结构画全，就会知道设备名称（sysName）的 Object Identifer 是 1.3.6.1.2.1.1.5

（iso.org.dod.internet.mgmt.mib-2.system.sysName），而设备位置（sysLocation）的 Object Identifer 是 1.3.6.1.2.1.1.6（iso.org.dod.internet.mgmt.mib-2.system.sysLocation）。

13.2.4 MIB 协议

SMI 协议只是定义了一系列的规则，MIB 才是这些规则在被管理的设备上的具体应用。在一个被管理的设备上，MIB 必须确定出对于本设备而言具体有哪些被管理的对象（Object）；MIB 必须给本设备中的每一个被管理对象确定出具体的名字；MIB 还必须给每一个被管理对象指定一个类型，如此等等。

概括地讲，MIB 的作用就是在被管理的设备上创建一个数据库，这个数据库中包含了若干个具有名字、具有类型、具有内容的被管理对象，这些对象的名字、类型等属性统统都必须遵从 SMI 规范。实质上，在这个数据库中，一个被管理对象（Object）就相当于是一个变量（Variable），被管理对象的名字就相当于是变量名，被管理对象的类型就相当是变量的类型，被管理对象的内容就相当于是变量的取值。

通常，我们把上面所描述的数据库称为管理信息库，或简称为 MIB（Management Information Base）。注意，这里的 MIB 并不是指 MIB 协议本身，而是指 MIB 协议应用在被管理的设备上之后而生成的那个数据库，如图 13-8 所示。MIB 数据库位于被管理的设备中，它是该设备中各种需要被管理的物理对象在数学意义上的集中反映。例如，当设备内部的温度下降时，MIB 中的某个变量的值就会减小；当设备的某个接口每收到一个 IP 报文时，MIB 中的另一个变量的值就会增加 1，如此等等。

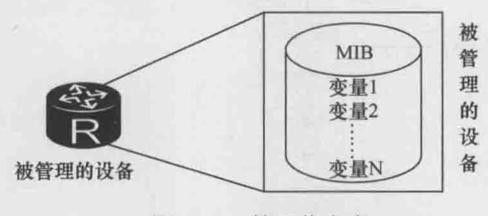


图 13-8 管理信息库 MIB

13.2.5 SNMP 协议

图 13-9 显示了 SNMP 协议的基本架构。在 SNMP 协议中，运行在网管员的电脑上的程序称为 Manager，也就是 SNMP Client；运行在被管理的设备上的程序称为 Agent，也就是 SNMP Server。SNMP 协议定义了若干种 SNMP 报文（例如，SNMPv3 定义了 8 种类型的 SNMP 报文，分别是 GetRequest、GetNextRequest、GetBulkRequest、SetRequest、Response、Trap、InformRequest、Report），并通过在 Manager 和 Agent 之间交换这些报文，从而实现管理信息的交流，进而实现网络管理的目的。

SNMP 报文都是封装在 UDP 报文中的。从 Manager 去往 Agent 的 SNMP 报文，其相应的 UDP 报文的目的端口号为 161；从 Agent 去往 Manager 的 SNMP 报文，其相应的 UDP 报文的目的端口号为 162。

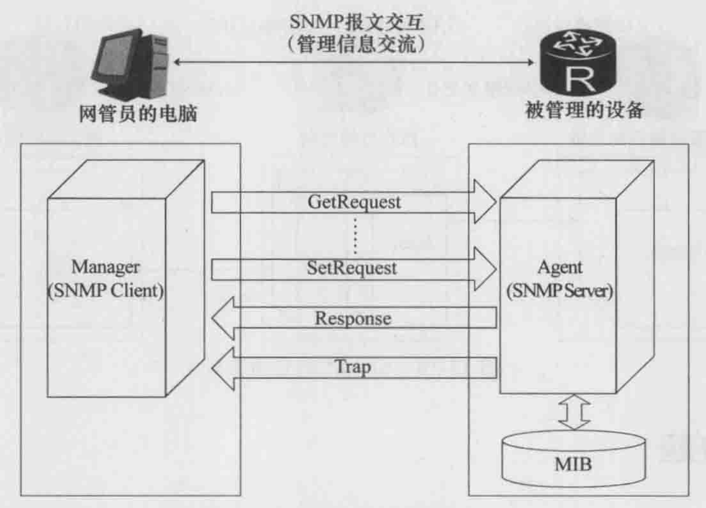


图 13-9　SNMP 的基本架构

如图 13-9 所示，Manager 需要查询被管理的设备上的某个被管理对象（Object）的信息时，可以向 Agent 发送一个 GetRequest 报文。Agent 接收到这个 GetRequest 报文后，会去 MIB 中提取出相应的 Object 的信息，并将这些信息封装在 Response 报文中，然后将 Response 报文发送给 Manager。

Manager 需要设置或修改被管理的设备上的某个被管理对象（Object）的信息时，可以向 Agent 发送一个 SetRequest 报文。Agent 接收到这个 SetRequest 报文后，会去 MIB 中对相应的 Object 的信息进行设置或修改。这样一来，便可实现 Manager 对于被管理对象的远程控制。

关于 Manager 对于被管理对象的远程控制，我们可以看看这样一个例子。假设某台服务器上有一个倒数计时器，该倒数计时器的值对应了 MIB 中的一个变量。当这个变量的值为 0（即倒数计时器的值为 0）时，服务器就会自动关机。如果网管员希望这台服务器立即关机，就只需向服务器上的 Agent 发送一个 SetRequest 报文，该报文的含义就是将 MIB 中与倒数计时器的值相对应的那个变量的值设置为 0。Agent 执行了这样的操作后，服务器就会立即自动关机。如果网管员希望这台服务器在两个小时后自动关机，就只需向服务器上的 Agent 发送一个 SetRequest 报文，该报文的含义是将 MIB 中与倒数计时器的值相对应的那个变量的值设置为 7 200 秒。Agent 执行了这样的操作后，服务器就会在两个小时后自动关机。

如上所述，Manager 可以主动向 Agent 发送 GetRequest、SetRequest 等报文，从而实现对各种管理信息的查询和修改。另一方面，Agent 也可以主动向 Manager 发送 Trap 报文，Trap 报文中携带了各种“告警信息”。Manager 在接收到这些告警信息后，便可以及时地采取相应的管理动作。

SNMP 的架构可以是分级的。如图 13-10 所示，一个 Manager 可以有它的上一级 Manager，一个 Manager 对于它的上一级 Manager 来说相当于是一个 Agent。

最后，需要说明的是，SNMP 所经历的版本有 SNMPv1、SNMPv2、SNMPv2c、SNMPv2u、SNMPv3。关于这些不同版本之间的差异，这里不再赘述。

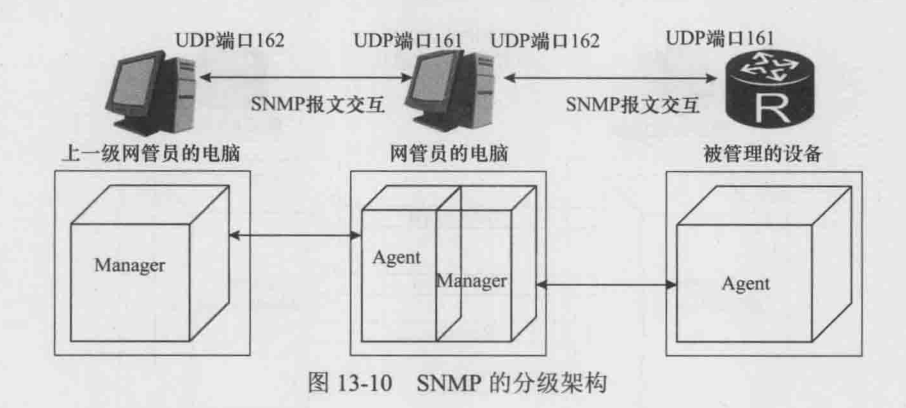


图 13-10　SNMP 的分级架构

13.3　练习题

1．（单选）下列选项中，哪一项才是一条合法的基本 ACL 的规则？（　）

A．rule permit ip　　B．rule deny ip

C．rule permit source any　　D．rule permit tcp source any

2．（单选）如果希望利用基本 ACL 来识别源 IP 地址为 172.16.10.0/24 网段的 IP 报文并执行“允许”的动作，那么应该采用下面哪一条规则？（　）

A．rule permit source 172.16.10.0 0.0.0.0

B．rule permit source 172.16.10.0 255.255.255.255

C．rule permit source 172.16.10.0 0.0.255.255

D．rule permit source 172.16.10.0 0.0.0.255

3．（单选）如果希望利用高级 ACL 来识别源 IP 地址为 172.16.10.1 且目的 IP 地址是 172.16.20.0/24 网段的 IP 报文并执行“拒绝”的动作，那么应该采用下面哪一条规则？（　）

A．rule deny source 172.16.10.1 0.0.0.0

B．rule deny source 172.16.10.1 0.0.0.0 destination 172.16.20.0 0.0.0.255

C．rule deny tcp source 172.16.10.1 0.0.0.0 destination 172.16.20.0 0.0.0.255

D．rule deny ip source 172.16.10.1 0.0.0.0 destination 172.16.20.0 0.0.0.255

4．（多选）关于高级 ACL 的规则，下列说法中正确的是？（　）

A．高级 ACL 的规则可用于识别报文的 TCP 目的端口号

B．高级 ACL 的规则可用于识别报文的 TCP 源端口号

C．高级 ACL 的规则可用于识别报文的 UDP 目的端口号

D．高级 ACL 的规则可用于识别报文的 UDP 源端口号

E．高级 ACL 的规则可用于识别报文的目的 IP 地址

5．（单选）网络管理系统主要涉及了哪 3 个协议？（　）

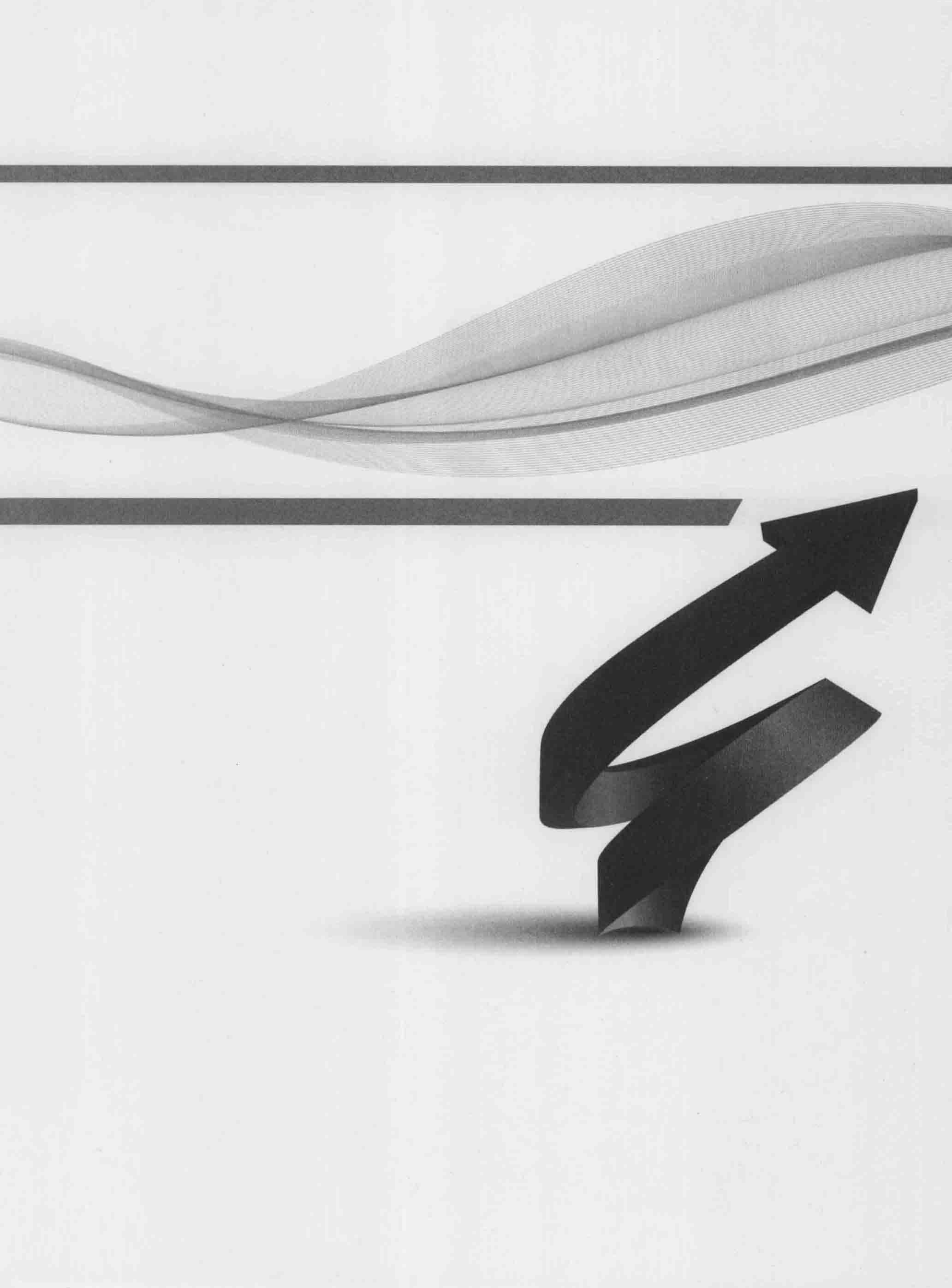
A．SMTP 协议、SMI 协议、MIB 协议

B．SNMP 协议、SMI 协议、RIP 协议

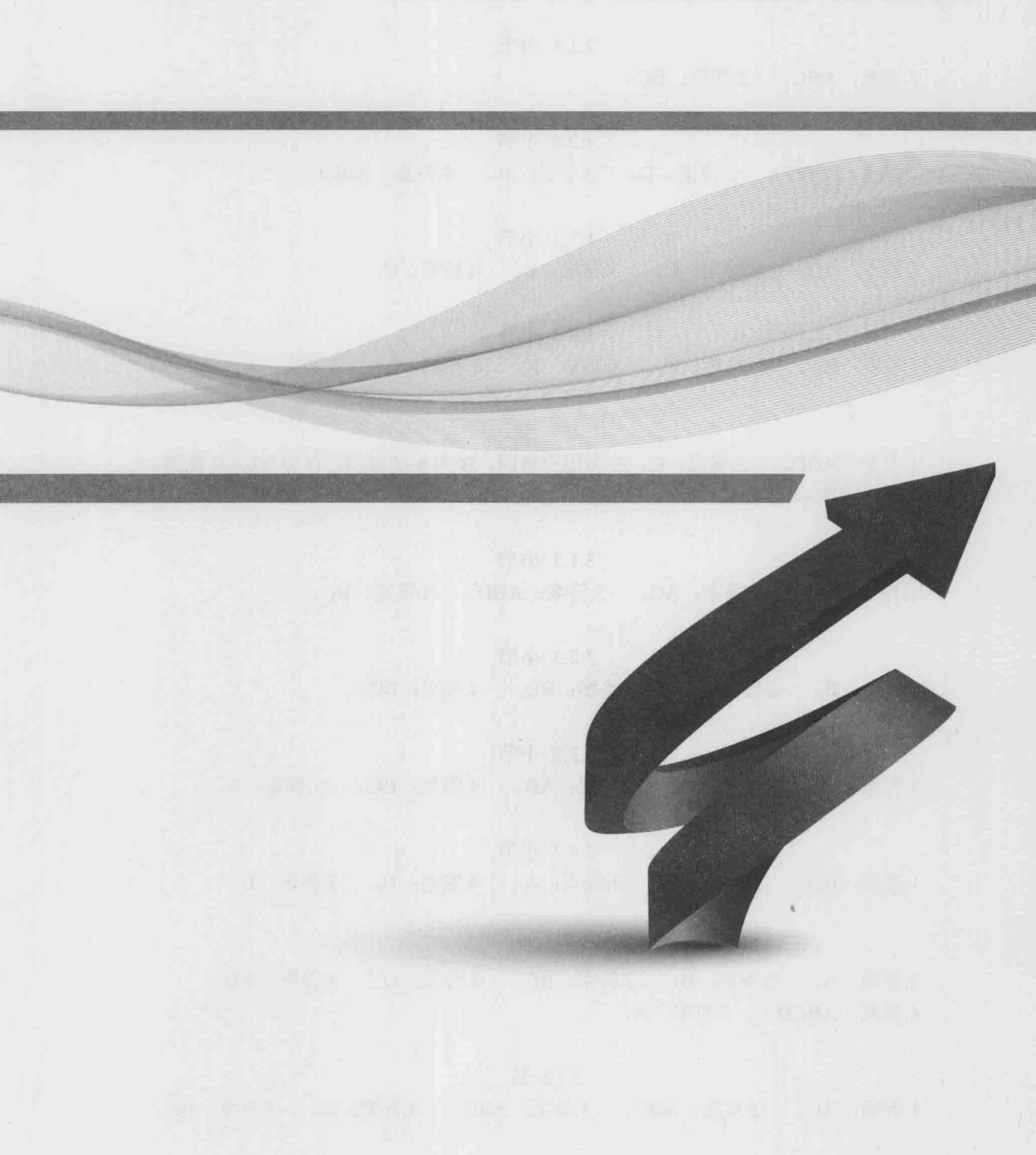
C．SNMP 协议、SMI 协议、MIB 协议

6．（单选）在 SNMP 协议中，Trap 报文的目的端口号是多少？（　）

A．161　　B．162　　C．163



附录
练习题答案



1.1.4 小节

1.答案：ABC。 2.答案：BC。

1.2.4 小节

1.答案：ABEF。 2.答案：D。 3.答案：B。 4.答案：ABCE。

1.3.3 小节

1.答案：AD。 2.答案：C。 3.答案：E。 4.答案：C。

1.4.3 小节

1.答案：BC。 2.答案：B。 3.答案：B。 4.答案：B。 5.答案：A。

2.9 节

1.答案：ABD。 2.答案：C。A为用户视图，B为系统视图，D为VLAN视图。
3.答案：B。 4.答案：C。 5.答案：ABCD。

3.1.3 小节

1.答案：BD。 2.答案：AC。 3.答案：ABD。 4.答案：BC。

3.2.3 小节

1.答案：B。 2.答案：B。 3.答案：BD。 4.答案：BC。

3.3.6 小节

1.答案：B。 2.答案：C。 3.答案：AD。 4.答案：BC。 5.答案：B。

3.4.3 小节

1.答案：BD。 2.答案：C。 3.答案：A。 4.答案：B。 5.答案：B。

4.7 节

1.答案：A。 2.答案：B。 3.答案：BC。 4.答案：D。 5.答案：AD。
6.答案：ABCD。 7.答案：A。

5.10 节

1.答案：AC。 2.答案：ABC。 3.答案：ABC。 4.答案：A。 5.答案：BC。

6.7 节

1.答案：ABD。 2.答案：CEG。 3.答案：A。 4.答案：C。 5.答案：D。
6.答案：AB。 7.答案：ABC。

7.4 节

1.答案：C。　2.答案：C。　3.答案：ABC。　4.答案：DE。
5.答案：CD。　6.答案：C。

8.1.9 小节

1.答案：C。　2.答案：ABC。　3.答案：A。　4.答案：B。
5.答案：D。　6.答案：A。

8.2.9 小节

1.答案：CD。　2.答案：C。　3.答案：D。　4.答案：D。
5.答案：ABD。　6.答案：AB。

8.3.12 小节

1.答案：B。　2.答案：ABCG。　3.答案：AB。　4.答案：C。
5.答案：CFH。　6.答案：AC。　7.答案：D。

9.5 节

1.答案：AC。　2.答案：BCD。　3.答案：AC。　4.答案：BCEFG。

10.4 节

1.答案：ABCF。　2.答案：AC。　3.答案：C。　4.答案：AB。　5.答案：E。

11.3 节

1.答案：ABCD。　2.答案：C。　3.答案：BC。　4.答案：AB。
5.答案：CDF。　6.答案：D。

12.3 节

1.答案：ABCD。　2.答案：AD。　3.答案：B。　4.答案：B。　5.答案：D。

13.3 节

1.答案：C。　2.答案：D。　3.答案：D。　4.答案：ABCDE。
5.答案：C。　6.答案：B。

封面设计：Dile迪利

本书是一本配套华为HCNA认证的学习指导用书，旨在帮助读者学习和理解HCNA网络技术原理知识中的要点和难点，内容包括：网络通信基础知识简介、华为VRP操作系统简介、以太网的工作原理、STP协议、VLAN原理、IP基础知识、TCP与UDP、路由协议基础、RIP协议、OSPF协议、VLAN间的三层通信、链路聚合技术、Smart Link与Monitor Link、DHCP、地址转换技术、PPP与PPPoE以及网络安全与网络管理。

本书对于HCNA网络技术原理知识中的要点和难点进行了非常详细而透彻的讲解，特别适合于学习和备考HCNA的读者朋友。对于希望准确而深刻地理解路由交换基础知识的高校学生、ICT从业人员以及网络技术爱好者，本书亦极具参考价值。





ISBN 978-7-115-38634-2

定价 :59.00 元

分类建议：计算机网络

人民邮电出版社网址：www.ptpress.com.cn